**Calhoun: The NPS Institutional Archive**

**DSpace Repository**

Theses and Dissertations | 1. Thesis and Dissertation Collection, all items

2009-03

# Dilution of Precision (DOP) calculation for mission planning purposes

Yuen, Ming Fatt.

Monterey, California. Naval Postgraduate School

http://hdl.handle.net/10945/4810

*Downloaded from NPS Archive: Calhoun*

http://www.nps.edu/library

Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

# NAVAL POSTGRADUATE SCHOOL

**MONTEREY, CALIFORNIA**

# THESIS

**DILUTION OF PRECISION (DOP) CALCULATION FOR MISSION PLANNING PURPOSES**

by

Ming Fatt Yuen

March 2009

| | |
|---|---|
| Thesis Advisor: | Morris R. Driels |
| Co-advisor: | Richard M. Harkins |

**Approved for public release; distribution is unlimited.**

THIS PAGE INTENTIONALLY LEFT BLANK

| REPORT DOCUMENTATION PAGE | | | Form Approved OMB No. 0704-0188 |
|---|---|---|---|
| Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503. | | | |
| **1. AGENCY USE ONLY** *(Leave blank)* | **2. REPORT DATE** March 2009 | **3. REPORT TYPE AND DATES COVERED** Master's Thesis | |
| **4. TITLE AND SUBTITLE** Dilution of Precision (DOP) Calculation for Mission Planning Purposes | | **5. FUNDING NUMBERS** | |
| **6. AUTHOR(S)** Ming Fatt Yuen | | | |
| **7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)** Naval Postgraduate School Monterey, CA 93943-5000 | | **8. PERFORMING ORGANIZATION REPORT NUMBER** | |
| **9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES)** N/A | | **10. SPONSORING/MONITORING AGENCY REPORT NUMBER** | |
| **11. SUPPLEMENTARY NOTES** The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. | | | |
| **12a. DISTRIBUTION / AVAILABILITY STATEMENT** Approved for public release; distribution is unlimited | | **12b. DISTRIBUTION CODE** | |


**13. ABSTRACT (maximum 200 words)**

The Joint Munitions Effectiveness Manuals (JMEM) were developed by the Joint Technical Coordinating Group for Munitions Effectiveness (JTCG/ME) to provide a set of data and methodologies that would permit a standardized comparison of weapon effectiveness across all service communities. In recent years, the JMEM are being integrated into a single software program that allows users to determine the effectiveness of weapon systems against a specified target irrespective of the weapon delivery mode. As part of the upgrading effort, this thesis aims to develop a program, written in Visual C++, to automate the calculation of the Dilution of Precision (DOP) associated with the delivery accuracy of GPS guided weapon systems. The DOP values generated by the program were compared with those generated by commercial DOP calculation software for validation. Relationship between the Vertical DOP and Horizontal DOP as well as the effect of using outdated almanac information to calculate DOP values were studied. It was found that the loss of one visible satellite could cause the DOP to increase by as much as 38%.


| **14. SUBJECT TERMS** GPS, DOP, JMEM, Delivery Accuracy | | | **15. NUMBER OF PAGES** 221 |
|---|---|---|---|
| | | | **16. PRICE CODE** |
| **17. SECURITY CLASSIFICATION OF REPORT** Unclassified | **18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified | **19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified | **20. LIMITATION OF ABSTRACT** UU |


NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

**Approved for public release; distribution is unlimited**

# DILUTION OF PRECISION (DOP) CALCULATION FOR MISSION PLANNING PURPOSES

Ming Fatt Yuen
Engineer, Singapore Defence Science & Technology Agency
B.Eng.(Hon.), National University of Singapore, 2003

Submitted in partial fulfillment of the
requirements for the degrees of

**MASTER OF SCIENCE IN MECHANICAL ENGINEERING**

**MASTER OF SCIENCE IN APPLIED PHYSICS**

from the

**NAVAL POSTGRADUATE SCHOOL**
**March 2009**

Author: Ming Fatt Yuen

Approved by: Morris R. Driels
Thesis Advisor

Richard M. Harkins
Co-Advisor

Knox T. Millsaps
Chairman, Department of Mechanical & Astronautical Engineering

James H. Luscombe
Chairman, Department of Physics

THIS PAGE INTENTIONALLY LEFT BLANK

# ABSTRACT

The Joint Munitions Effectiveness Manuals (JMEM) were developed by the Joint Technical Coordinating Group for Munitions Effectiveness (JTCG/ME) to provide a set of data and methodologies that would permit a standardized comparison of weapon effectiveness across all service communities. In recent years, the JMEM are being integrated into a single software program that allows users to determine the effectiveness of weapon systems against a specified target irrespective of the weapon delivery mode. As part of the upgrading effort, this thesis aims to develop a program, written in Visual C++, to automate the calculation of the Dilution of Precision (DOP) associated with the delivery accuracy of GPS guided weapon systems. The DOP values generated by the program were compared with those generated by commercial DOP calculation software for validation. Relationship between the Vertical DOP and Horizontal DOP as well as the effect of using outdated almanac information to calculate DOP values were studied. It was found that the loss of one visible satellite could cause the DOP to increase by as much as 38%.

THIS PAGE INTENTIONALLY LEFT BLANK

# TABLE OF CONTENTS


**I. INTRODUCTION....1**
**A. GLOBAL POSITIONING SYSTEM (GPS) ....1**
**1. Overview of the NAVSTAR Global Positioning System ....1**
***a. Space Segment....1***
***b. Control Segment....2***
***c. User Segment....2***
**2. Workings of GPS ....2**
**3. Sources of Errors in GPS ....3**
***a. Atmospheric Effect....4***
***b. Multipath....5***
***c. Ephemeris and Clock....5***
***d. Relativity....6***
***e. Sagnac Distortion....6***
***f. Selective Availability....6***
***g. Jamming....7***
***h. Number of Visible Satellites ....7***
**4. Dilution of Precision ....8**
**B. JOINT MUNITIONS EFFECTIVENESS MANUALS (JMEM)....11**
**1. Introduction to JMEM ....11**
**2. Use of JMEM in Mission Planning....11**
**3. Effect of DOP on Mission Planning....12**
**C. MOTIVATION OF THESIS ....12**
**D. OBJECTIVES OF THESIS ....13**

**II. THEORY ....15**
**A. POSITIONS OF GPS SATELLITES ....15**
**1. Almanac Data....15**
**2. Calculations ....17**
***a. Gregorian to Julian Date Conversion....17***
***b. Satellite Position in ECEF Frame....18***
***c. ECEF to ENU Conversion ....20***
**B. DOP CALCULATIONS....21**
**1. Identification of Visible Satellites....21**
**2. DOP Calculations....22**

**III. IMPLEMENTATION ....25**
**A. APPROACH....25**
**B. CODING WITH MATLAB AND VISUAL C++ ....25**
**C. CODE TESTING ....27**
**1. Matlab Code Testing....27**
**2. Visual C++ Code Testing....28**

**IV. RESULTS ....30**
**A. COMPARISON OF CALCULATED DOP VALUES....30**

1. Matlab vs Trimble....30
2. Visual C++ vs Trimble....32

V. ANALYSIS/ DISCUSSION OF RESULTS....35
A. COMPARISON OF CALCULATED DOP VALUES....35
1. Difference in Number of Visible Satellites....35
*a. Leap Seconds....35*
*b. Visibility Algorithm....36*
2. No Difference in Number of Visible Satellites....37
3. Effect of One Visible Satellite on DOP....37
B. EFFECT OF USING OUTDATED ALMANAC DATA....38
1. Approach....38
2. No Difference in Number of Visible Satellites....38
3. Difference in Number of Visible Satellites....38
C. AVERAGE HDOP AND VDOP....39
1. Approach....39
2. Results....40

VI. CONCLUSION....43
A. FUTURE WORK....44
1. Integration of Code into JWS....44
2. Enhancement of GUI....44

APPENDIX A. EXAMPLES OF ALMANAC FILE....45
A. EXAMPLE OF ALMANAC FILE IN SEM (.AL3) FORMAT....45
B. EXAMPLE OF ALMANAC FILE IN YUMA (.ALM) FORMAT....53

APPENDIX B. VISUAL C++ CODES FOR DOP CALCULATION....67
A. DOP_CALCULATORDLG.CPP....67
B. DOP_CALCULATORDLG.H....102
C. GRAPH.CPP [AFTER 20]....105
D. GRAPH.H [AFTER 20]....143
E. MATRIX.H [AFTER 21]....148

APPENDIX C. MATLAB CODES FOR DOP CALCULATION....153

APPENDIX D. DOP COMPARISON BETWEEN MATLAB & TRIMBLE....159
A. VDOP....159
B. TDOP....161
C. HDOP....163
D. PDOP....165
E. GDOP....167
F. DOP COMPARISON IN TABLE....169

APPENDIX E. DOP COMPARISON BETWEEN VISUAL C++ & TRIMBLE 181
A. VDOP....181
B. TDOP....182
C. HDOP....183
D. PDOP....184

**E. GDOP ..... 185**
**F. DOP COMPARISON IN TABLE ..... 186**

**APPENDIX F. COMPARISON OF VISIBLE SATELLITES BETWEEN MATLAB & TRIMBLE ..... 189**

**APPENDIX G. COMPARISON OF DOP VALUES FROM OUTDATED ALMANAC DATA ..... 193**

**APPENDIX H. AVERAGE DOP VALUES ..... 195**

**APPENDIX I. COMPARISON BETWEEN VDOP & HDOP FOR OBSTRUCTION ANGLE = $10^{O}$ ..... 197**

**LIST OF REFERENCES ..... 199**

**INITIAL DISTRIBUTION LIST ..... 201**

THIS PAGE INTENTIONALLY LEFT BLANK

# LIST OF FIGURES


Figure 1. Earth Centered, Earth Fixed (ECEF) Frame [After 1] ..... 2
Figure 2. Effect of Satellite Geometry on Dilution of Precision [After 13] ..... 9
Figure 3. Schematic for Identifying Visible Satellite ..... 22
Figure 4. Graphical User Interface of the DOP Calculator ..... 26
Figure 5. GDOP generated by Matlab and Trimble Program at Various Positions on Earth ..... 32
Figure 6. GDOP generated by Visual C++ and Trimble Program at Various Positions on Earth ..... 33

THIS PAGE INTENTIONALLY LEFT BLANK

# LIST OF TABLES


Table 1. Meaning of DOP Values [From 4].....10
Table 2. Useful Information in Almanac File (SEM Format) [After 5] .....16
Table 3. Test Parameters for Matlab Code .....28
Table 4. Test Parameters for Visual C++ Code .....28
Table 5. Global Average VDOP and HDOP.....40

THIS PAGE INTENTIONALLY LEFT BLANK

# ACKNOWLEDGMENTS

I would like to thank Professor Morris Driels for his continuous guidance throughout all phases of the thesis.

I would also like to give special thanks to my fellow schoolmates, Mr. Koh Chuan Lian and Mr. Ong Kim Soo, DSTA, Singapore as well as Mr. Vincent Cheong, ST, Singapore for their assistance on C++ programming. Without their assistance, my task to develop the DOP calculator program in Visual C++ would have been much more difficult.

THIS PAGE INTENTIONALLY LEFT BLANK

# I. INTRODUCTION

## A. GLOBAL POSITIONING SYSTEM (GPS)

### 1. Overview of the NAVSTAR Global Positioning System

The NAVSTAR Global Positioning System (GPS) is a satellite-based navigation and positioning system made up of a constellation of between 24 to 32 satellites or space vehicles (SV). The GPS was developed by the United States Department of Defense and although the GPS was originally intended for military applications, the United States government made the system freely available for civilian use in the 1980s. GPS works in any weather conditions, anytime and anywhere in the world.

The system consists of three segments: the space segment, the control segment, and the user segment. The United States Air Force maintains and operates the space and control segments.

#### *a. Space Segment*

The space segment consists of the orbiting satellites. The GPS constellation has a minimum of 24 satellites traveling on six medium Earth orbits (altitude about 20,200 km) of approximately 55° inclination (tilt relative to Earth's equator) and are separated by 60° right ascension of the ascending node (angle along the equator from a reference point to the orbit's intersection). Each satellite completes one orbit in slightly less than 12 hours.

Each satellite transmits its own unique microwave signals on two different L-band frequencies that give information on the precise orbit for the satellite sending the message (the ephemeris); the approximate orbits and general health of all satellites (the almanac); as well as an ionospheric delay model. All satellites broadcast at the same two frequencies, 1.57542 GHz (L1 signal) and 1.2276 GHz (L2 signal). The receiver can distinguish the signals from different satellites because they are encoded with a pseudo-random number (PRN) sequence that is different for each satellite. The receiver knows the PRN codes for each satellite and uses this to reconstruct the navigation message.

### *b. Control Segment*

The control segment consists of five monitor stations around the Earth that maintain the satellites in their proper orbits through occasional maneuvers, and adjust the satellite clocks. It tracks the satellites, uploads updated navigational data, and maintains health and status of the satellite constellation.

### *c. User Segment*

The user segment consists of the GPS receiver equipment, which receives the signals from the satellites and uses the transmitted information to calculate the receivers' positions, velocities and headings based on the positions of the satellites and the time the signal was transmitted and received.

## 2. Workings of GPS

Information on the positions of the satellites is transmitted by the satellites and can be calculated relative to a set of coordinates that are Earth centered, Earth fixed (ECEF) (see Figure 1).

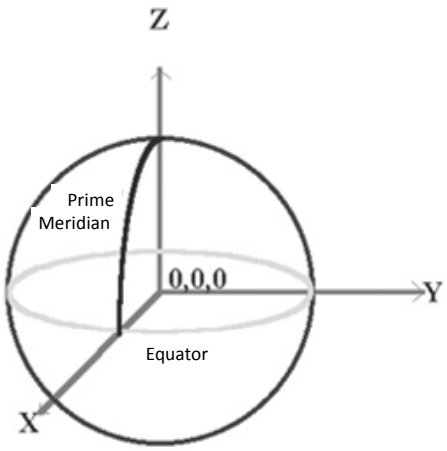



Figure 1. Earth Centered, Earth Fixed (ECEF) Frame [After 1]

The range, $R$, between the GPS receiver and each satellite is measured by timing the delay between the transmission time of the signal from the satellite and the arrivaltime of the same signal to the receiver, $\Delta t$ i.e., the signal travel time. Since these signals travel at the speed of light, $c$, the range between the GPS receiver and the satellite is given by

$$R = c \times \Delta t \tag{1.1}$$

Three coordinates in the ECEF frame define the GPS receiver's position. This implies that three different range measurements would suffice to determine the position of the receiver. However, clocks used in GPS receivers are not as accurate as the atomic clocks in the satellites. As an error of a nanosecond in time measurement would result in an error of about 0.3 m in range calculation, each range measurement needs to be corrected to account for the receiver clock's inaccuracies. As these measured ranges are distorted by the relatively inaccurate time keeping of the receiver's clock, they are known as pseudoranges.

Since the GPS receiver clock error is an unknown variable in addition to the three position coordinates of the GPS receiver, a minimum of four range measurements, instead of three, are required to resolve the GPS time and determine the position of the receiver using trilateration.

Trilateration can be described as using the pseudoranges to form spheres around the respective satellites such that the position of the receiver lies within the overlapping region of the spheres. It should be noted that trilateration is different from triangulation in that trilateration uses ranges, while triangulation uses angles to determine the position of a point.

## 3. Sources of Errors in GPS

The above description of how GPS works assumed that there are no other sources of error other than the GPS receiver clock. In reality, there are many sources, which can introduce error into the calculation of the GPS receiver position. These errors need to be accounted for in order to mitigate their effects on the accuracy. The sources of errors include the following:

### *a. Atmospheric Effect*

As the satellite signal passes through the atmosphere, its speed is reduced as air has a slightly higher index of refraction (about 1.0003 [2]) causing the signal speed to decrease by an average of about 0.03%. However, inconsistencies in the atmosphere, especially the ionosphere, cause the signal to slow in a non-uniform manner. This effect is least when the satellite is directly overhead and become greater for satellites near the horizon since the signal path through the atmosphere is longer.

In order to mitigate this effect, after the receiver's approximate position is known using the pseudoranges, a built-in mathematical model can be used to estimate the average amount of delay to compensate for this type of error. However, the model may not be able to predict the full effects of the ionospheric delay. A more accurate way to compensate for this error is to use both frequencies to measure the time delay. Ionospheric delay affects the speed of microwave signals differently depending on their frequency, this is a characteristic known as dispersion. Delays measured on two frequency bands can be used to measure dispersion, and this measurement can then be used to estimate the delay at each frequency.

Another way to compensate for the ionospheric error is to compare the GPS-measured position with a known surveyed position. This takes advantage of the fact that the effects of the ionosphere generally change slowly and can be averaged over time; hence, the correction on the ionospheric error can be applied to other GPS receivers in the same general region. Satellite Based Augmentation Systems (SBAS) such as WAAS (available in North America and Hawaii), EGNOS (Europe and Asia) or MSAS (Japan) transmits the ionospheric correction data via satellite, while Ground Based Augmentation Systems (GBAS) transmits the correction data via ground radio transmitter directly to the GPS receiver.

Humidity in the troposphere also results in errors similar to ionospheric delay. However, this effect is more localized, changes more quickly than ionospheric effects, and is not frequency dependent. These characteristics make tropospheric effects

more difficult to measure and compensate compared to ionospheric effects. Typically, error in pseudorange caused by ionospheric effects are about $\pm 5$ m while the troposheric effect is about $\pm 0.5$ m [3].

### *b. Multipath*

When the GPS signal is reflected off objects such as tall buildings, the travel time of the signal before it reaches the receiver increases. This results in multipath errors. Various techniques have been developed to mitigate multipath errors. For long delay multipath, the receiver itself can be programmed to recognize the wayward signal and discard it. For shorter delay multipath from the signal reflected off the ground, specialized antennas may be used to reduce the power received by the antenna from such reflected signals. Short delay reflections are harder to distinguish from routine fluctuations in atmospheric delay.

Multipath effects are less severe in moving vehicles as solutions using reflected signals quickly fail to converge and only the direct signals result in stable solutions. Typically, error in pseudorange caused by multipath effect is about $\pm 1$ m [3].

### *c. Ephemeris and Clock*

The satellites transmit ephemeris data (data on their precise orbits) every 30 seconds, but the data itself may be up to two hours old. Although data up to four hours old is considered valid, it may not indicate the satellite's actual position.

The satellite's atomic clocks encounter noise and clock drift errors. While the navigation message contains corrections for these errors and estimates of the accuracy of the atomic clock, they are based on observations done at the monitor stations and may not indicate the clock's actual state. However, these errors are typically small. Error in pseudorange caused by ephemeris error is about $\pm 2.5$ m while clock error is about $\pm 2$ m [3].

### *d. Relativity*

According to the theory of relativity, the clocks on the satellites are affected by their speed (special relativity) as well as their gravitational potential (general relativity). Due to the weaker gravitational field at the GPS orbit, general relativity predicts that time speeds up by about 45.9 μs per day. On the other hand, special relativity predicts that time slows by about 7.2 μs per day due to the orbital speed of the satellite. Hence, the total effect is that time on the satellite speeds up by about 39 μs per day.

Since accurate time keeping is central to the accuracy of GPS, this discrepancy has to be accounted for and this is done by giving the frequency standard on board each satellite a rate offset prior to launch, making it run slightly slower than the desired frequency on Earth; specifically, at 10.22999999543 MHz instead of 10.23 MHz.

### *e. Sagnac Distortion*

Sagnac distortion is caused because GPS time is defined in an inertial frame while observations are processed in an ECEF frame. A Lorentz transformation is applied to convert from the inertial frame to the ECEF frame and the resulting correction on the signal travel time has opposite algebraic signs for satellites in the Eastern and Western celestial hemispheres. Although the effect is small, neglecting it will produce an east-west error of about a few hundreds of nanoseconds, or tens of meters in position [3].

### *f. Selective Availability*

Selective Availability (SA) is a feature in GPS that, when enabled, can introduce intentional random errors of up to a hundred meters into the civilian navigation signals with the intention of limiting accurate positioning capability to the United States military and other authorized users.

However, this feature was turned off in May 2000 following an executive order from the United States President Bill Clinton to set the SA error to zero by 2006. This allowed civilian applications, such as the aviation industry, to take advantage of the

highly accurate navigation signals. In Sep 2007, the United States Department of Defense announced that future GPS satellites would no longer support SA, thereby making the policy permanent.

### *g. Jamming*

Having travelled about 20,200 km from the satellites to Earth, GPS signals received by GPS receivers on Earth tend to be relatively weak. Hence, it is easy for other sources of electromagnetic (EM) radiation to overpower the GPS signals, making acquiring and tracking the satellite signals difficult or impossible. These sources of EM radiation can occur naturally or made artificially.

An example of naturally occurring EM radiation capable of disrupting GPS reception is solar flare. Solar flares are explosions in the Sun that produces strong EM radiation that has the potential to disruption satellite communication. Other examples include naturally occurring geomagnetic storms, found mainly near the poles of the Earth's magnetic field as well as interference from Van Allen Belt radiation when the satellites pass through the South Atlantic Anomaly.

An example of artificial source is man-made jammers, which typically emit strong EM radiation to overpower the actual GPS signal. These signals can interfere with GPS receivers when they are within radio range or line of sight.

### *h. Number of Visible Satellites*

The GPS constellation is designed to have at least six satellites above any part of the Earth at any one time. However, reception of the signals from these satellites may be blocked by nearby obstacles such as buildings, terrain and dense vegetation making position calculations less accurate. In the worst case, all signals may be blocked making position calculation impossible. In general, the higher the number of visible satellites, the better the accuracy.

## 4. Dilution of Precision

The dilution of precision (DOP) also contributes to the accuracy of the GPS calculations but not in a direct manner. Mathematically, DOP is the ratio between the standard deviations of a specified parameter and the pseudorange. For example, Vertical DOP is the ratio between the standard deviation of the vertical component (altitude) of the GPS receiver and the standard deviation of the pseudorange. For parameters that involve more than one variable such as Geometric DOP, the ratio is between the root sum square of the standard deviation of the variables (x, y, z coordinates and time) and the standard deviation of the pseudorange.

Physically, DOP describes the geometric strength of the visible satellites' configuration on the GPS accuracy. Ideally, the visible satellites should be located at wide angles relative to each other. The geometry of such satellite configuration is said to be strong and the DOP values are low. Conversely, if the visible satellites have small angular separation, the satellites' configuration has weak geometry and the DOP values are high.

Figure 2 shows a scenario where a GPS receiver measures the pseudoranges of two satellites. Keeping the error of the range measurement constant in both cases, the case on the left, with larger angular separation between the two satellites, shows that the area of uncertainty on the position of the receiver is smaller than the case on the right. The DOP value, hence, may be understood as the 'dilution' factor on the accuracy of the original measurement.

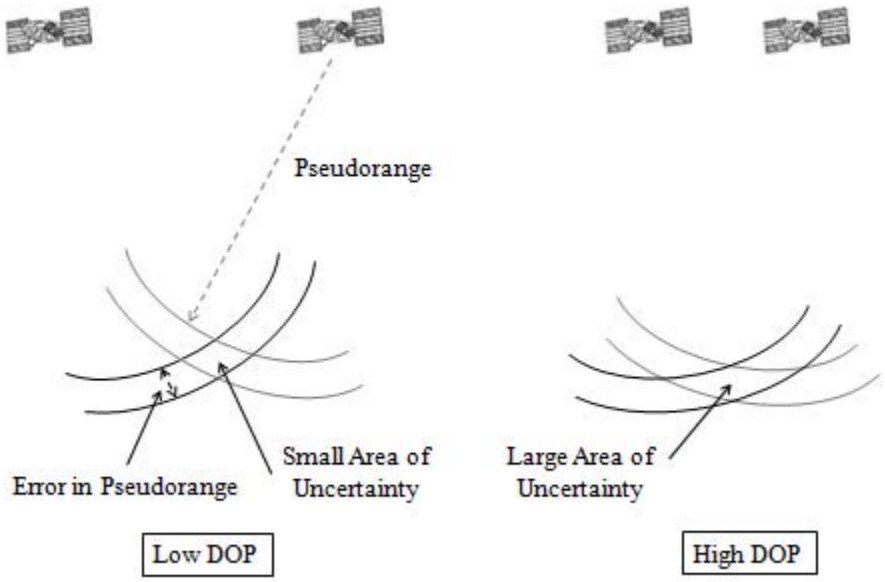



Figure 2. Effect of Satellite Geometry on Dilution of Precision [After 13]

Table 1. Meaning of DOP Values [From 4]

| DOP Value | Rating | Description |
|---|---|---|
| 1 | Ideal | This is the highest possible confidence level to be used for applications demanding the highest possible precision at all times. |
| 2-3 | Excellent | At this confidence level, positional measurements are considered accurate enough to meet all but the most sensitive applications. |
| 4-6 | Good | Represents a level that marks the minimum appropriate for making business decisions. Positional measurements could be used to make reliable in-route navigation suggestions to the user. |
| 7-8 | Moderate | Positional measurements could be used for calculations, but the fix quality could still be improved. A more open view of the sky is recommended. |
| 9-20 | Fair | Represents a low confidence level. Positional measurements should be discarded or used only to indicate a very rough estimate of the current position. |
| 21-50 | Poor | At this level, measurements are inaccurate by as much as 300 meters with a 6-meter accurate device and should be discarded. |

Table 1 gives a description and the meaning for various DOP values. It should be noted that DOP values of less than 1 are possible i.e., the accuracy of the calculated position (via trilateration) could be higher than the accuracy of the pseudoranges measured by the GPS receiver. When there are sufficient visible satellites with wide angular separation, it is possible for the region of uncertainty on the position of the

receiver to be reduced to the point that it is smaller than the uncertainty of the individual pseudorange measurement. This is inherent in the trilateration method.

## B. JOINT MUNITIONS EFFECTIVENESS MANUALS (JMEM)

### 1. Introduction to JMEM

The Joint Technical Coordinating Group for Munitions Effectiveness (JTCG/ME) produced the Joint Munitions Effectiveness Manuals (JMEM) in order to provide a set of data and methodologies that standardize weapon effectiveness calculations, thereby facilitating comparisons of weapon effectiveness between different communities in the military.

JMEM include detailed information on the physical characteristics and performance of weapons and weapon systems; descriptions of the mathematical methodologies that employ these data to generate effectiveness estimates; software that permit users to calculate effectiveness estimates; and pre-calculated weapon effectiveness estimates. They are used by all services in United States as well as NATO and other allies to plan operational missions, support training and tactics development, and support force-level analyses.

In the past, JMEM were volumes of orange covered manuals but over time, computer programs were used to supplement the manuals to allow faster computation as well as more realistic (but also more complex and computationally intensive) models to be used. As computers became more affordable and widely used, the paper version of JMEM gave way to CD versions. Beginning in 2007, all JMEM weapon effectiveness products are integrated into a single program called the JMEM Weaponeering System (JWS). This is a target-oriented program, which allows users to determine the effectiveness of weapon systems against a specified target regardless of the weapon delivery mode.

### 2. Use of JMEM in Mission Planning

During the planning of offensive missions, it is important for military planners to know about the delivery accuracy as well as the effectiveness of selected weapon system

against a specified target. This will allow the planners to estimate the number of weapon systems required to destroy the target.

The effectiveness of a weapon system against a specified target is quantified by the effectiveness index or lethal area. This is dependent on factors such as the defined kill criterion; the physical and geometrical configuration of the target and its critical components; nature of the weapon system; and, the damage required on the critical component(s) to achieve the desired kill criterion.

The delivery accuracy of a weapon system, on the other hand, is quantified by the distribution of the weapon system's impact points such as deflection error probable (DEP) and range error probable (REP). The weapon system's accuracy is dependent on many factors such as target acquisition error, tracking error, etc.

### 3. Effect of DOP on Mission Planning

For the case of GPS-guided weapon systems, the error associated with GPS in determining the position of the target and the weapon is one of the most important sources of errors to take into account when calculating the delivery accuracy of the weapon system. As DOP has the effect of amplifying the original GPS measurement errors, any mission planners who intend to use GPS-guided weapon systems would need to know the DOP.

Since DOP is derived from the configuration of the visible satellites, it varies depending on the time and position of the target. For mission planners, that variation in DOP could make a difference to the effectiveness of an offensive mission and the potential collateral damage.

## C. MOTIVATION OF THESIS

As part of the effort to integrate the JMEM into a single software program, this thesis aims to develop a program to automate the calculation of DOP associated with the delivery accuracy of GPS-guided weapon systems.

## D. OBJECTIVES OF THESIS

This thesis aims to develop a program in Visual C++ to automate the calculation of DOP using the almanac file in SEM format (.al3). In addition, the thesis also aims to study the effect of the using outdated almanac data in the DOP calculation and find out the relationship between the Horizontal DOP and Vertical DOP.

THIS PAGE INTENTIONALLY LEFT BLANK

# II. THEORY

## A. POSITIONS OF GPS SATELLITES

The position of satellite at any given instant can be calculated from the ephemeris of the satellite. The almanac is a practical and convenient source to get the ephemeris of all the satellites in the constellation. Although the almanac only gives the rough ephemeris, the accuracy is good enough for calculating DOP values. Moreover, the almanacs are posted ahead of time, thereby allowing planning to be done.

### 1. Almanac Data

The almanacs are available to the public from the United States Coast Guard website, http://www.navcen.uscg.gov/GPS/almanacs.htm, in the form of a file that is updated almost daily. The almanac file is available in two formats, namely SEM (.al3) and YUMA (.alm). Examples of an almanac file in SEM and YUMA format are shown in Appendix A.

The YUMA format is more reader-friendly, but the SEM format is more compact. Hence, in view of efficiency, the SEM format was chosen as the input file to calculate the positions of all the satellites at any given time. Useful information contained in each SEM almanac file is shown in

Table 2. It should be noted that angles are given in the terms of number of semicircles, hence it is important to convert them to radians before proceeding with further calculations.

Table 2. Useful Information in Almanac File (SEM Format) [After 5]

| Information | Unit | Description |
|---|---|---|
| Number of Records | records | The number of satellite almanac records contained in the file |
| GPS Week Number, *WN* | weeks | The almanac reference week for all almanacs in the file as per ICD-GPS-200 |
| GPS Time of Applicability, *TOA* | sec | The almanac reference time for all almanacs in the file as per ICD-GPS-200 |
| PRN Number | none | The satellite PRN number as per ICD-GPS-200. Used to identify individual satellite |
| Eccentricity, $e$ | unitless | The satellite almanac orbital "eccentricity" as defined in ICD-GPS-200 |
| Inclination Offset, $\delta i_k$ | semicircles | The satellite almanac orbital "inclination angle offset" as defined in ICD-GPS-200 |
| Rate of Right Ascension, $\dot{\Omega}$ | semicircles/ sec | The satellite almanac orbital "rate of right ascension" as defined in ICD-GPS-200 |
| Square root of Semi-Major Axis, $\sqrt{A}$ | $m^{1/2}$ | The satellite almanac orbital "square root of the semi-major axis" as defined in ICD-GPS-200 |
| Longitude of Orbital Plane, $\Omega_0$ | semicircles | The satellite almanac orbital "geographic longitude of the orbital plane at the weekly epoch" as defined in ICD-GPS-200 |
| Argument of Perigee, $\omega$ | semicircles | The satellite almanac orbital "argument of perigee" as defined in ICD-GPS-200 |
| Mean Anomaly at Reference Time, $M_0$ | semicircles | The satellite almanac orbital "mean anomaly" as defined in ICD-GPS-200 |
| Satellite Health | none | The satellite health code expressed in integer form |

## 2. Calculations

### *a. Gregorian to Julian Date Conversion*

Before information extracted from the almanacs can be used to calculate the positions of the satellites, the date and time needs to be specified and converted to the same form as those in the almanac, namely the GPS week number (the number of weeks since Jan 6, 1980- the reference start date for GPS) and number of seconds of that week. This is done by subtracting the specified date with the GPS reference start date. However, this is difficult to do using the Gregorian calendar (the internationally accepted calendar used today, for example, Mar 27, 2009). Hence, the dates are processed in Julian dates instead. The conversion from Gregorian to Julian date, *JD* is shown below [6]

$$JD = 367 \cdot Y - floor(\frac{7}{4}(Y + floor(\frac{M+9}{12})) - floor(\frac{3}{4}(floor(\frac{Y + \frac{M-9}{7}}{100}) + 1)$$

$$+ floor(\frac{275 \cdot M}{9}) + D + 1721028.5 + \frac{hr}{24} + \frac{\min}{1440} + \frac{\sec}{86400}$$

$$- \frac{timezone + daylightsaving}{24} + \frac{leap\,\sec}{86400} \qquad (1.2)$$

In the above equation, "Y", "M", "D", "hr", "min" and "sec" represents the year, month, day, hours, minutes and seconds of the specified local date and time respectively; "timezone" represents the number of hours offset from GMT or Zulu Time, for example, the offset for Eastern Standard Time (North America) is -5 hours; "daylightsaving" represents an additional hour when daylight saving is in effect; "leap sec" represents the number of leap seconds added since Jan 6, 1980. The "floor" operator returns the integer value.

Using the above conversion, the Julian date for Jan 6, 1980 is 2,444,244.5 days. The number of weeks since Jan 6, 1980, *NumWeek*, is therefore

$$NumWeek = floor(\frac{JD - 2444244.5}{7}) \qquad (1.3)$$

If *NumWeek* is more than 1024, minus *NumWeek* by 1024 until it is less than 1024. This is due to the GPS week rollover issue [7].

Finally, the number of seconds of that *NumWeek*, *NumSec*, is given by

$$NumSec = (JD - 2444244.5 - 7 \cdot NumWeek) \cdot 86400 \tag{1.4}$$

***b. Satellite Position in ECEF Frame***

With the time specified, the positions of the satellites can then be calculated with reference to the ECEF frame. There are two constants required in the calculation, namely the WGS 84 value of the Earth's Universal Gravitational Parameter, μ = 3.986005 × $10^{14}$ $m^3/sec^2$ and the WGS 84 value of the Earth's Rotation Rate, $\dot{\Omega}_e$ = 7.2921151467 × 10-5 rad/sec. Using data from the almanac file for each satellite, the following parameters are calculated sequentially to obtain the satellite position in the ECEF frame [8].

The Computed Mean Motion, $n_o$ is given by

$$n_0 = \sqrt{\frac{\mu}{A^3}} \tag{1.5}$$

The Time since *TOA*, $t_k$ is given by

$$t_k = (NumWeek - WN) \cdot 604800 + (NumSec - TOA) \tag{1.6}$$

The Mean Anomaly, $M_k$ is given by

$$M_k = M_0 + n_0 t_k \tag{1.7}$$

The Kelper's Equation for Eccentric Anomaly is shown below where $E_k$ needs to be solved by iteration since the expression for $E_k$ is not explicit.

$$E_k = M_k + e \cdot \sin E_k \tag{1.8}$$

The True Anomaly, $\upsilon_k$ is calculated using the value of $E_k$ obtained from Equation 2.7 as follows.

$$\upsilon_k = \tan^{-1}\{\frac{(\sqrt{1-e^2}\sin E_k)/(1-e\cdot\cos E_k)}{(\cos E_k - e)/(1-e\cdot\cos E_k)}\} \tag{1.9}$$

The Eccentric Anomaly, $E_k$ is then recalculated using $\upsilon_k$ obtained from Equation 2.8 and this new value is used for subsequent equations.

$$E_k = \cos^{-1}\{\frac{e+\cos\upsilon_k}{1+e\cdot\cos\upsilon_k}\} \tag{1.10}$$

The Corrected Argument of Latitude, $u_k$ is given by

$$u_k = \upsilon_k + \omega \tag{1.11}$$

The Corrected Radius, $r_k$ is given by

$$r_k = A(1-e\cdot\cos E_k) \tag{1.12}$$

The Corrected Inclination, $i_k$ is given by

$$i_k = i_0 + \delta i_k \tag{1.13}$$

The Satellite Position in Orbital Plane is given by

$$x_k^{'} = r_k \cos u_k \tag{1.14}$$

$$y_k^{'} = r_k \sin u_k \tag{1.15}$$

The Corrected Longitude of Ascending Node, $\Omega_k$ is given by

$$\Omega_k = \Omega_0 + (\dot{\Omega} - \dot{\Omega}_e)t_k - \dot{\Omega}_e(TOA) \tag{1.16}$$

Finally, the Satellite Position in ECEF Frame is given by

$$x_k = x_k^{'}\cdot\cos\Omega_k - y_k^{'}\cdot\cos y_k \sin\Omega_k \tag{1.17}$$

$$y_k = x_k^{'}\cdot\sin\Omega_k + y_k^{'}\cdot\cos y_k \cos\Omega_k \tag{1.18}$$

$$z_k = y_k^{'}\cdot\sin i_k \tag{1.19}$$

### *c. ECEF to ENU Conversion*

In order to obtain the Horizontal DOP value with reference to the Earth's surface, the positions of the satellites need to be converted to the East-North-Up (ENU) coordinates relative to a local reference point specified on the Earth surface. This position corresponds to the latitude ($\phi$) and longitude ($\lambda$) of the GPS receiver's position.

There are two constants required in the calculation, namely the WGS 84 value of the Earth's Semi-Major Axis, $a$ = 6,378,137 m and the WGS 84 value of the Earth's First Eccentricity, $e1$ = $8.181919084266 \times 10^{-2}$. Furthermore, the local reference point is expressed in ECEF coordinates to simplify the ECEF to ENU conversion for the satellites' positions. The conversion from geodetic ($\lambda, \phi$, and altitude ($alt$)) to ECEF coordinates (x, y, z) is shown below [9]. It should be noted that for the local reference point, alt is zero since it is specified to be on the Earth's surface.

The Prime Vertical Radius of Curvature, $N$ is given by

$$N = \frac{a}{\sqrt{1 - e^2 \sin^2 \phi}} \tag{1.20}$$

The conversion from Geodetic to ECEF Coordinates is

$$x = (N + alt)\cos\phi\cos\lambda \tag{1.21}$$

$$y = (N + alt)\cos\phi\sin\lambda \tag{1.22}$$

$$z = ((1 - e^2)N + alt)\sin\phi \tag{1.23}$$

The conversion from ECEF to ENU coordinates is shown below. The subscript $obj$ in the ECEF coordinates represents the object of interest (in our case, it can be the satellite or GPS receiver), while $l$ represents the local reference point.

$$Est = -(x_{obj} - x_l)\sin\lambda + (y_{obj} - y_l)\cos\lambda \tag{1.24}$$

$$Nth = -(x_{obj} - x_l)\sin\phi\cos\lambda - (y_{obj} - y_l)\sin\phi\sin\lambda + (z_{obj} - z_l)\cos\phi \tag{1.25}$$

$$Up = (x_{obj} - x_l)\cos\phi\cos\lambda + (y_{obj} - y_l)\cos\phi\sin\lambda + (z_{obj} - z_l)\sin\phi \tag{1.26}$$

## B. DOP CALCULATIONS

### 1. Identification of Visible Satellites

After knowing the positions of all the satellites at the given time, the visible satellites need to be identified. For ease of calculation, the GPS receiver's position is converted from the geodetic coordinates (the GPS receiver's position is entered by the user in geodetic coordinates) to the ECEF coordinates by using Equation 2.20 - 2.22.

In addition, several assumptions are made. Firstly, it is assumed that line of sight is needed for the satellite's signal to be visible to the GPS receiver. Secondly, the Earth is assumed a perfect sphere. This differs from the WGS 84 model by about 1 in 300 parts and it is assessed to be a reasonable assumption. Thirdly, the GPS receiver is assumed to be relatively close to the surface of the Earth such that the field of view of the sky above the receiver is constant regardless of its altitude.

The method to identify visible satellites is illustrated in Figure 3. The ECEF coordinates of the GPS receiver and satellite are represented by vectors originating from the center of the Earth, $\overrightarrow{OT}$ and $\overrightarrow{OS}$ respectively. Any obstruction to the field of view of the entire sky above the receiver, such as terrain, is represented by the angle $\beta$. $\alpha$, the angle between $\overrightarrow{OT}$ and $\overrightarrow{TS}$ (where $\overrightarrow{TS} = \overrightarrow{OS} - \overrightarrow{OT}$), is given by

$$\alpha = \cos^{-1}\left(\frac{\overrightarrow{OT}\cdot\overrightarrow{TS}}{\left|\overrightarrow{OT}\right|\left|\overrightarrow{TS}\right|}\right) \tag{1.27}$$

Hence, the satellite is visible to the GPS receiver if $\alpha$ is less than $90^{o}$- $\beta$.

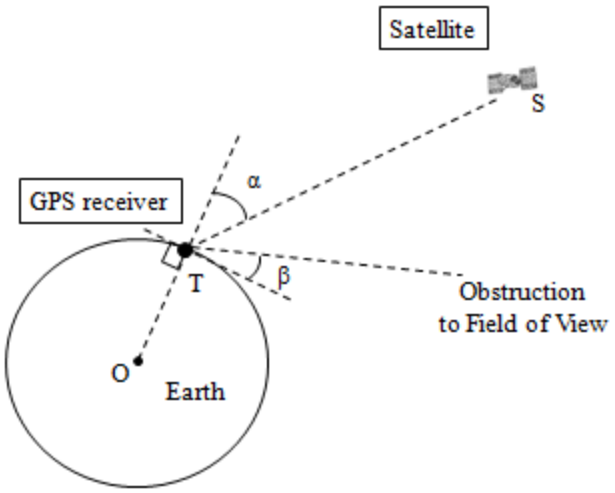



Figure 3. Schematic for Identifying Visible Satellite

## 2. DOP Calculations

With the visible satellites identified, the pseudoranges between the GPS receiver and the visible satellites, $\rho_i$ are calculated as shown below. The subscript *i* represents the numbering of each visible satellite, while *r* represents the GPS receiver. Since the ENU frame is defined from the local reference point, $Est_r$ and $Nth_r$ are zero, while $Up_r$ is the altitude of the receiver.

$$\rho_i = \sqrt{(Est_i - Est_r)^2 + (Nth_i - Nth_r)^2 + (Up_i - Up_r)^2} \tag{1.28}$$

The directional derivatives of East, North, Up and Time in the ENU frame for each satellite are then calculated as shown below.

$$D_{Est_i} = \frac{Est_i - Est_r}{\rho_i} \tag{1.29}$$

$$D_{Nth_i} = \frac{Nth_i - Nth_r}{\rho_i} \tag{1.30}$$

$$D_{Up_i} = \frac{Up_i - Up_r}{\rho_i} \tag{1.31}$$

$$D_{t_i} = -1 \tag{1.32}$$

The directional derivatives of all the visible satellites form a (n × 4) matrix, $D$, where $n$ is the total number of visible satellites.

$$D = \begin{pmatrix} D_{Est_1} & D_{Nth_1} & D_{Up_1} & D_{t_1} \\ D_{Est_2} & D_{Nth_2} & D_{Up_2} & D_{t_2} \\ \vdots & \vdots & \vdots & \vdots \\ D_{Est_n} & D_{Nth_n} & D_{Up_n} & D_{t_n} \end{pmatrix} \tag{1.33}$$

Taking the inverse of $D^T \cdot D$ yields a (4 × 4) matrix where the diagonal terms gives the square of the East DOP (XDOP), North DOP (YDOP), Vertical DOP (VDOP) and Time DOP (TDOP) as shown below where the off-diagonal terms are not shown for clarity.

$$\left(D^T \cdot D\right)^{-1} = \begin{pmatrix} XDOP^2 & & & \\ & YDOP^2 & & \\ & & VDOP^2 & \\ & & & TDOP^2 \end{pmatrix} \tag{1.34}$$

The other DOP values, namely Horizontal DOP (HDOP), Position DOP (PDOP) and Geometric DOP (GDOP) are obtained by root summing the appropriate diagonal terms in $(D^T \cdot D)^{-1}$ as shown below [1].

$$HDOP = \sqrt{XDOP^2 + YDOP^2} \tag{1.35}$$

$$PDOP = \sqrt{XDOP^2 + YDOP^2 + VDOP^2} \tag{1.36}$$

$$GDOP = \sqrt{XDOP^2 + YDOP^2 + VDOP^2 + TDOP^2} \tag{1.37}$$

THIS PAGE INTENTIONALLY LEFT BLANK

# III. IMPLEMENTATION

## A. APPROACH

Matlab was selected as the software to test the algorithm of the DOP calculator program during development as Matlab is relatively easy to learn and manipulate compared to Visual C++. DOP results generated by the algorithm in Matlab were compared with DOP calculations generated by commercial DOP calculation software to check the algorithm.

The validated algorithm was then used to write the program in the Visual C++ environment and checked again with the results generated by the commercial DOP calculation software to ensure that the algorithm was implemented correctly in Visual C++. In addition, the option to plot graphs of various DOP values over a 24-hour period was added to enhance the usability of the program.

## B. CODING WITH MATLAB AND VISUAL C++

There are significant differences between the coding language in Matlab and Visual C++. For example, matrix manipulation is integrated into the Matlab language, while Visual C++ is more generic and requires additional codes to be written in order to carry out matrix operations. The same is true for graph plotting.

As such, instead of writing C++ codes for matrix manipulation and graph plotting from scratch, open source C++ codes available from the Internet are modified and incorporated into the DOP calculation program. The Visual C++ codes for DOP calculation are shown in Appendix B with credits given to the parts of code that were modified from open sources, while the Matlab codes are shown in Appendix C.

Figure 4 shows the screen shot of the Graphical User Interface (GUI) produced by the Visual C++ codes. The user inputs are grouped into "Time Inputs" and "Position Inputs" in the upper part of the window. The time inputs refer to the local time, while the position inputs refer to the position of the GPS receiver at which the DOP values are to be calculated.

Pressing the "Update Almanac" button will download the latest version of the almanac file from http://www.navcen.uscg.gov/GPS/almanacs.htm, which is the United States Coast Guard website, to the folder containing the rest of the C++ codes. The codes stating the web address and location of the downloaded file are in the C++ file "DOP_CalculatorDlg.cpp" line 230 and line 168 respectively.

The outputs are displayed in the lower part of the window after the "Compute DOP" button is pressed. The left portion shows the various calculated DOP values for the specified time inputs, while the right shows the graph that plots the DOP values over a 24-hour period on the specified day. By default, the HDOP is plotted but other DOP values can also be plotted by selecting the appropriate radio buttons.

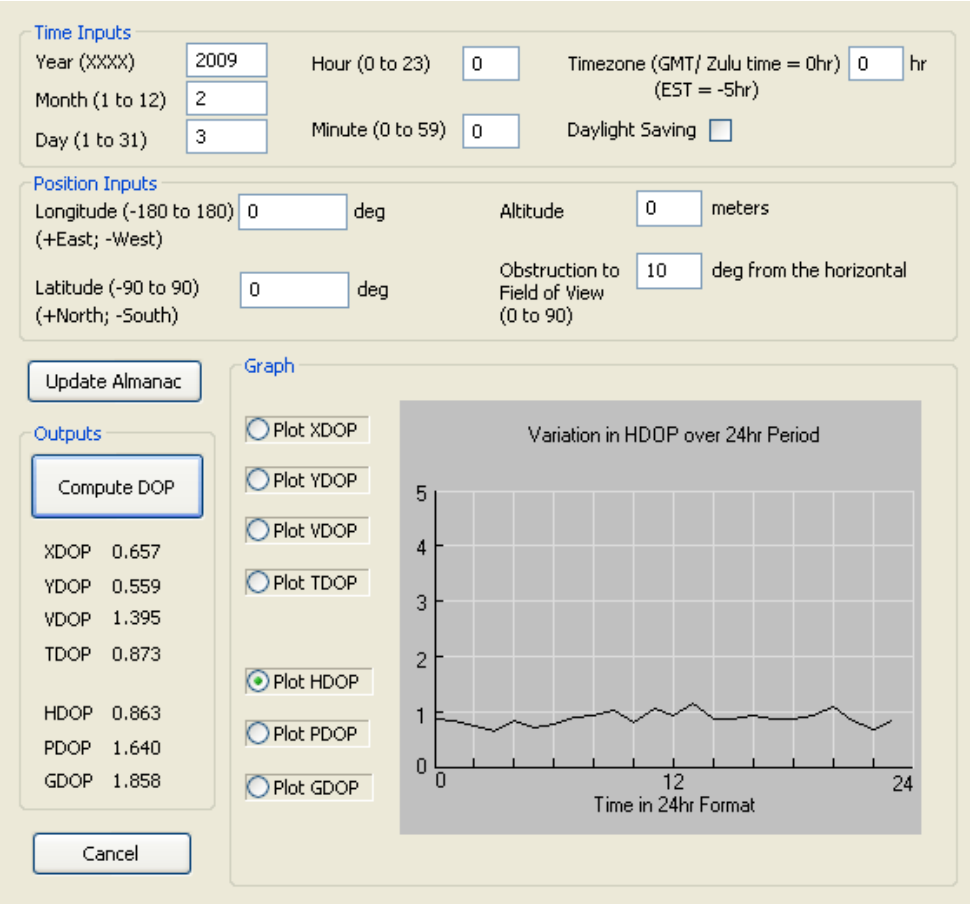



Figure 4. Graphical User Interface of the DOP Calculator

## C. CODE TESTING

DOP values generated by the program written in Matlab and Visual C++ were compared with results generated by the commercial software: GPS planning software version 2.74 from Trimble Navigation Limited, which is publically available at its website [10]. Trimble's planning software was chosen for a couple of reasons.

Firstly, it is the only free software found to accept the same input requirements and provide direct comparison for the various DOP values as its output. Secondly, Trimble is a listed company that deals with GPS applications, giving some level of credence to the accuracy of the results generated by the software.

Only GDOP, PDOP, HDOP, TDOP and VDOP were compared, as these are the five DOP values generated by the Trimble planning software. The results generated by the Trimble, Matlab and Visual C++ program were presented on graphs and tables for comparison.

Note that during program development, results generated by individual sections of the codes, such as the calculation of the position of satellite in the ECEF frame, time conversion from Gregorian to Julian date, etc, were also checked against external references [1], [6], [11]. Hence, the combination of checking individual sections and the entire program with different external references ensures that the DOP calculation algorithm was implemented correctly.

### 1. Matlab Code Testing

Two hundred sixty six combinations of latitude and longitude were sampled to test the codes written in Matlab. The test parameters are as follows:

Table 3. Test Parameters for Matlab Code

| Parameter | Value |
|---|---|
| Date and Time | Feb 3, 2009, 0000hr GMT |
| Position of GPS Receiver in Latitude | $-90^o$ (South) to $90^o$ (North) at $15^o$ interval |
| Position of GPS Receiver in Longitude | $-165^o$ (West) to $180^o$ (East) at $15^o$ interval |
| Altitude of GPS Receiver | 0 m |
| Obstruction to Field of View, $\beta$ | $\beta = 10^\circ$ |

### 2. Visual C++ Code Testing

The test parameters for the Visual C++ codes are similar to those for the Matlab codes, except that there were less data points as extracting DOP values from Visual C++ is more laborious. 62 combinations of latitude and longitude were sampled and the test parameters are as follows:

Table 4. Test Parameters for Visual C++ Code

| Parameter | Value |
|---|---|
| Date and Time | Feb 3, 2009, 0000hr GMT |
| Position of GPS Receiver in Latitude | $-90^o$ (South) to $90^o$ (North) at $30^o$ interval |
| Position of GPS Receiver in Longitude | $-150^o$ (West) to $180^o$ (East) at $30^o$ interval |
| Altitude of GPS Receiver | 0 m |
| Obstruction to Field of View, $\beta$ | $\beta = 10^\circ$ |

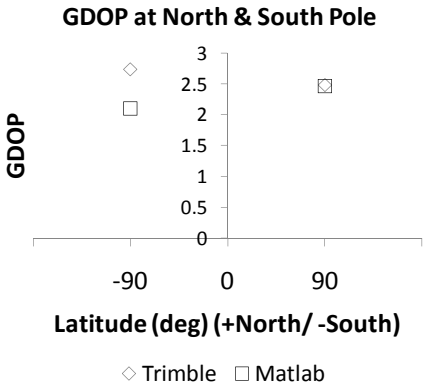
THIS PAGE INTENTIONALLY LEFT BLANK

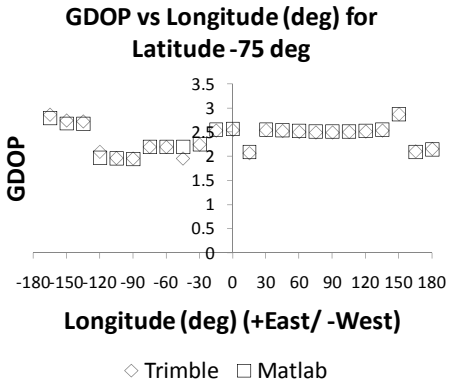
# IV. RESULTS

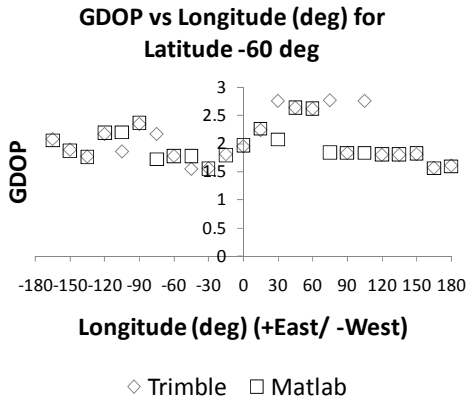
## A. COMPARISON OF CALCULATED DOP VALUES

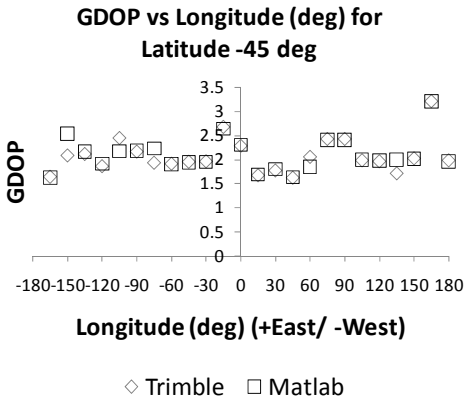
### 1. Matlab vs Trimble

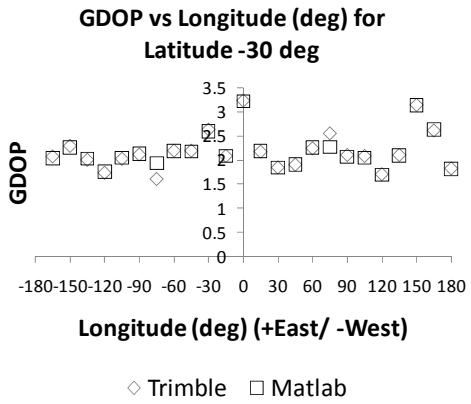
Of the 266 combinations of latitude and longitude sampled around the Earth, most of the differences between the DOP values generated by the Matlab and Trimble program were less than 5%. However, 39 of the samples (about 15% of the samples) gave differences that were between 5% and 38%. The graphs of the GDOP values are shown in Figure 5 below. Although there were differences in the percentage difference for various DOP values, GDOP gives a good representation on the trend since it is derived from the rest of the DOP values. The complete comparison of all the DOP values is shown in Appendix D.

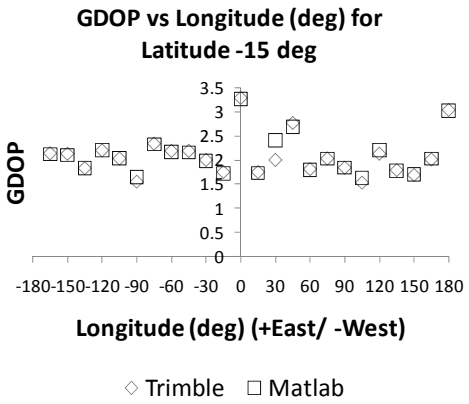


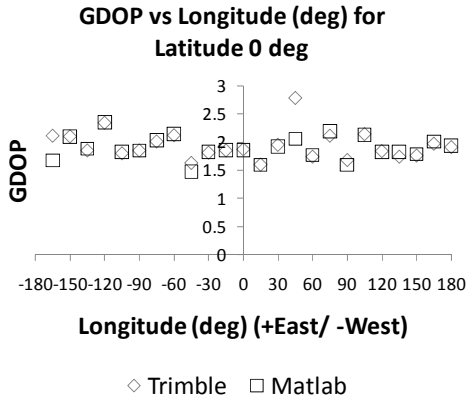



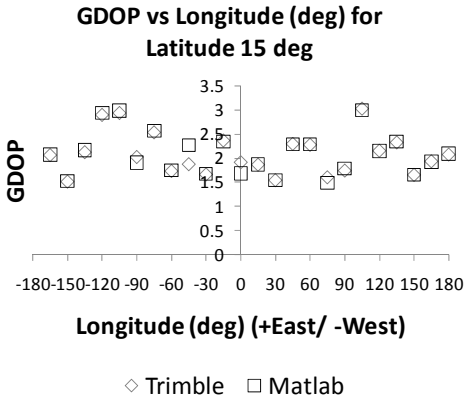

GDOP vs Longitude (deg) for Latitude -60 deg
GDOP
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab


GDOP vs Longitude (deg) for Latitude -45 deg
GDOP
3.5
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab


GDOP vs Longitude (deg) for Latitude -30 deg
GDOP
3.5
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab


GDOP vs Longitude (deg) for Latitude -15 deg
GDOP
3.5
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab


GDOP vs Longitude (deg) for Latitude 0 deg
GDOP
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab


GDOP vs Longitude (deg) for Latitude 15 deg
GDOP
3.5
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

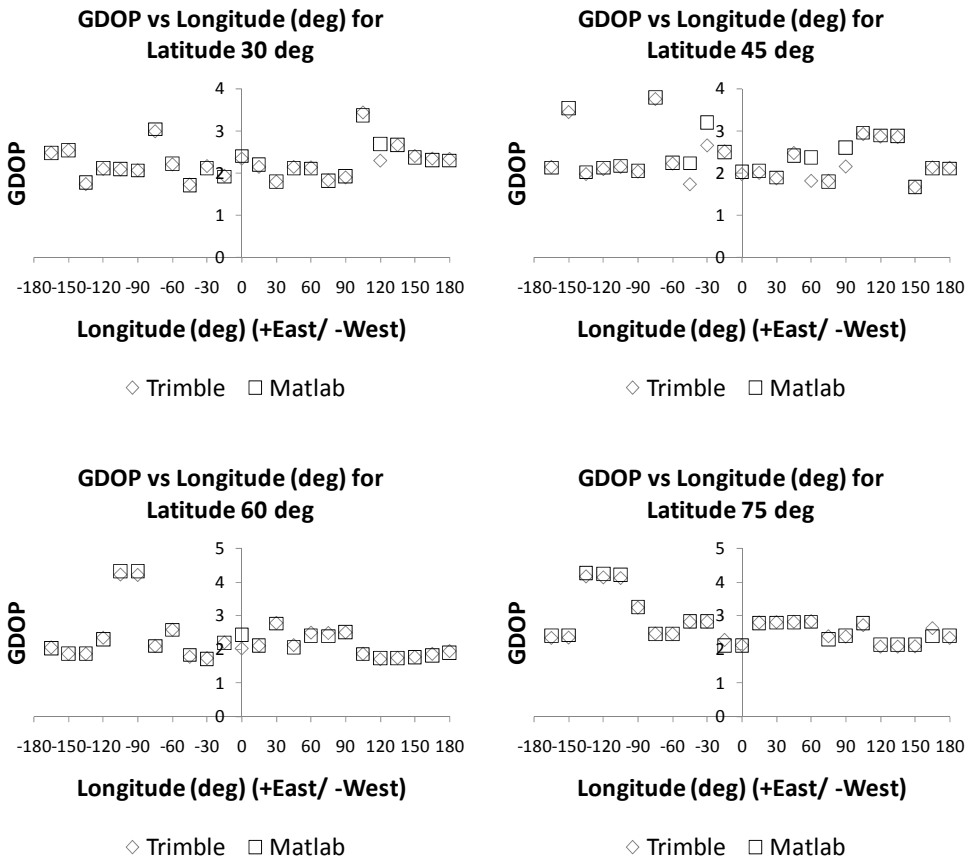



Figure 5. GDOP generated by Matlab and Trimble Program at Various Positions on Earth

## 2. Visual C++ vs Trimble

From the 62 samples, most of the differences between the DOP values generated by Visual C++ and Trimble program were also less than 5%. The sample points that gave more than 5% difference were the same as those between the Matlab and Trimble program. This is expected since the algorithm in the Visual C++ program is the same as the one in Matlab and the test samples for the Visual C++ program are a subset of those for the Matlab program. The graphs of the GDOP values are shown in Figure 6 below, while the complete comparison of all the DOP values is shown in Appendix E.

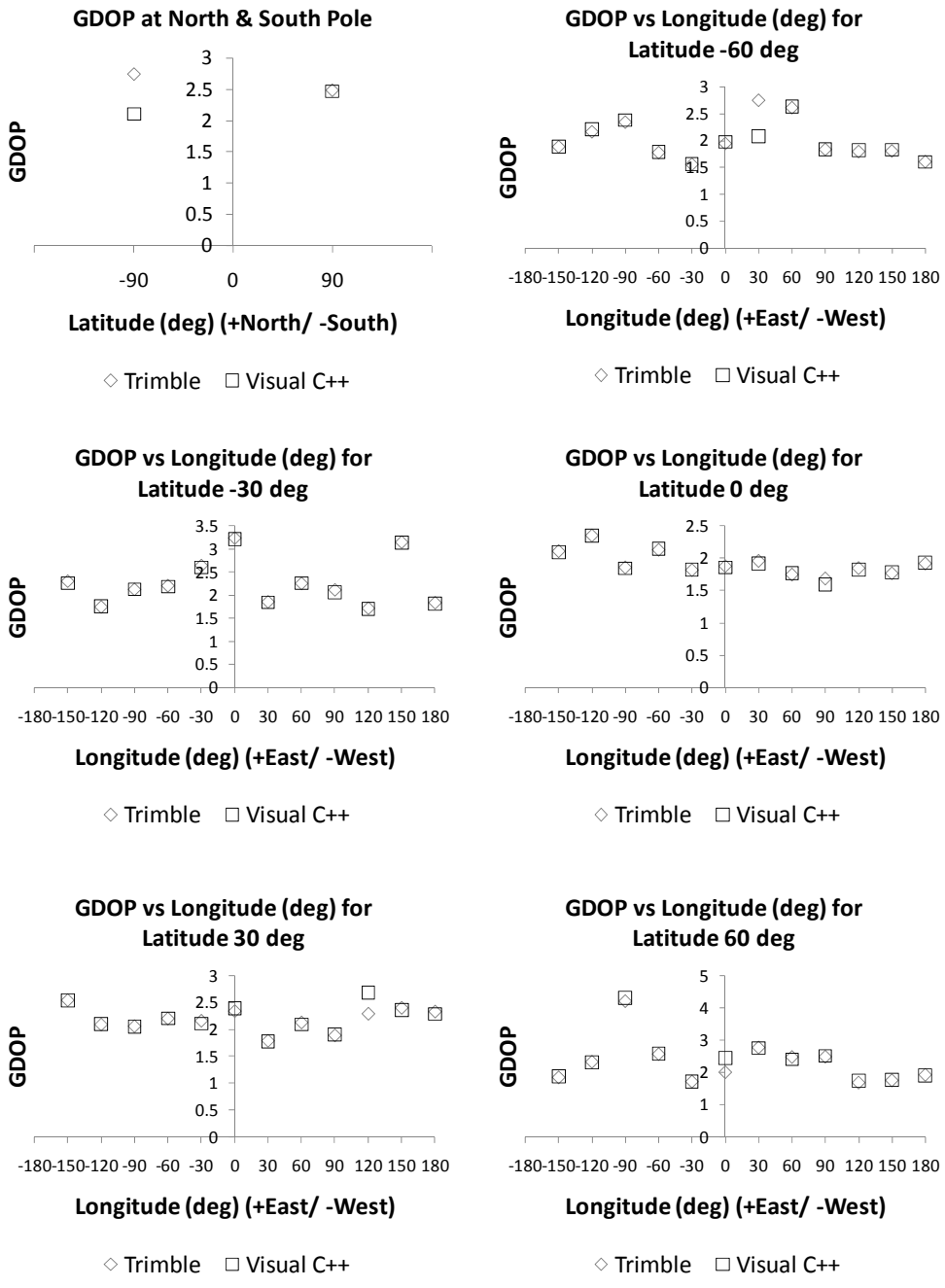



Figure 6. GDOP generated by Visual C++ and Trimble Program at Various Positions on Earth

THIS PAGE INTENTIONALLY LEFT BLANK

# V. ANALYSIS/ DISCUSSION OF RESULTS

## A. COMPARISON OF CALCULATED DOP VALUES

Since the results generated by the Visual C++ program are a subset of those generated by Matlab, analysis on the results generated by the Matlab and Trimble program will also be valid for the Visual C++ program. Hence, the analysis would focus on comparing the results from the Matlab and Trimble program only.

As stated in the previous chapter, for the DOP values generated by the Matlab and Trimble program, 39 samples had differences greater than 5%. It was found that the GPS receiver in 37 of the samples had different number of satellites in its field of view, while two samples had the same number of visible satellites.

### 1. Difference in Number of Visible Satellites

In the 37 samples, all of them had the number of visible satellites differed by one. In every case, the satellite in question was found to be near the visibility criterion. The visibility criterion is defined by the obstruction angle, $\beta$. In this case, $\beta$ was $10^o$, which meant that only satellites that were $10^o$ above the horizon were considered visible. In other words, only satellites that made an angle of less than $80^o$ with the GPS receiver (with respect to the Up-axis in ENU frame; see angle $\alpha$ in Figure 3) were in the field of view of the GPS receiver. Although angle $\alpha$, calculated for each satellite by the Matlab and Trimble program, were slightly different (differences were all within $4^o$), they were enough to cause one satellite to be included or excluded from the field of view of the GPS receiver in these samples.

#### *a. Leap Seconds*

It was found that in one of the 37 samples, this discrepancy could be accounted for by taking the effects of leap seconds into account. When two leap seconds were removed from the Matlab calculations (in the time conversion algorithm to convert to GPS time), the difference in the number of visible satellites of that sample (Latitude - $90^o$ (South), Longitude $0^o$) disappeared. For this sample point, satellite number 11 went

from visible to invisible in the Matlab calculation and this changed the DOP values such that the differences compared to the results from the Trimble program decreased from about 24% to less than 4%. As the source code for the Trimble program is not available, it is speculated that this version of the Trimble program was developed before 2005 and as a result, the two additional leap seconds, which were added on Jan 1, 2005 and Jan 1, 2009 [12], were not accounted for.

### *b. Visibility Algorithm*

For the other 36 samples, when the number of visible satellites were deliberately made to be the same as those seen by the Trimble program by adjusting the obstruction angle, $\beta$, the differences were found to drop to less than 5%. Hence, the discrepancies in these samples could be attributed to the difference in the algorithm used by the Trimble and Matlab program to determine the visibility of satellites. It is possible it is due to different assumptions made on the visibility algorithm.

Interestingly, it was found that one of the samples (Latitude $60^o$ (North), Longitude $90^o$ (East): not among the 37 samples mentioned earlier) had a difference of less 2% in the DOP values although the number of visible satellites also differed by one. On closer examination, it was found that there were two satellites (satellite number 15 and 26) close to each other that were near the visibility criterion. Since DOP is dependent on the angular separation of the visible satellites, the loss of one of these two satellites from the field of view caused little change to the DOP. In this case, the angular separation between the two satellites was about $18^o$. Satellite number 26 was in the field of view in the Trimble program, while it was just out of it in the Matlab program.

Details on the comparison of the angles of the visible satellites are shown in Appendix F. Note that the angles that caused the discrepancies are shaded in grey for clarity.

### 2. No Difference in Number of Visible Satellites

Having accounted for the large DOP percentage differences in 37 of the 39 samples, the remaining two samples (Latitude -45$^{o}$ (South), Longitude 60$^{o}$ (East) & Latitude 0$^{o}$, Longitude 90$^{o}$ (East)) had the same number of visible satellites from both programs.

On closer examination, it was found that although the total number of visible satellites was the same, the actual satellites seen by the GPS receiver were different. By sheer coincidence, one of the satellites was considered visible in the Trimble program but not in Matlab program, while another satellite was considered visible in the Matlab program but not in the Trimble program.

For the sample point Latitude -45$^{o}$ (South), Longitude 60$^{o}$ (East), satellite number 26 was considered visible by the Trimble program but was marginally out of the visibility criterion in the Matlab program, while the reverse is true for satellite number 23. This caused the DOP values to differ by about 15%.

For Latitude 0$^{o}$, Longitude 90$^{o}$ (East), satellite number 11 was considered visible by the Trimble program but invisible by the Matlab program, while the reverse is true for satellite number 26. This resulted in about 7% difference in DOP values.

### 3. Effect of One Visible Satellite on DOP

The effect of an addition or loss of a visible satellite (or different visible satellite for the case where the total number of visible satellites is the same) on the DOP value is dependent on the angular separation between that satellite and the rest of the visible satellites. If none of the remaining visible satellites were near that satellite, the effect of seeing or losing sight of the additional satellite near the visibility criterion would make a disproportionally big difference to the DOP values. The greatest difference in the DOP values was found to be about 38% due to a loss of one visible satellite (Latitude -60$^{o}$ (South), Longitude 75$^{o}$ (North) & Latitude -60$^{o}$ (South), 105$^{o}$ (East), where there were 9 other visible satellites remaining in the field of view).

Conversely, if there were visible satellites near that particular satellite, the effect of seeing or losing sight of the additional satellite would be small. This is evident in the sample point Latitude $60^o$ (North), Longitude $90^o$ (East) having a difference of less than 2% in the DOP values although the number of visible satellite also differed by one.

## B. EFFECT OF USING OUTDATED ALMANAC DATA

The almanac file is updated almost daily. However, there might be circumstances where the most updated almanac file could not be accessed. As such, the effect of using outdated almanac data on DOP values was studied.

### 1. Approach

In the study, an arbitrary date was chosen and the almanac file was used as the benchmark. The chosen date in this study was Jul 28, 2008 (Almanac file: Week 466, Time of Applicability 319488). Using this almanac file, the DOP values on dates that were 1, 7, 14 and 30 days later were calculated. The position of the GPS receiver was fixed at Latitude $0^o$, Longitude $0^o$ at altitude 0 m and the obstruction angle, $\beta$ was set at $10^o$. These DOP values were then compared with the values obtained from using the correct almanac files for the respective dates.

### 2. No Difference in Number of Visible Satellites

It was found that using outdated almanac data had little effect on the DOP values if the total number of visible satellites remained the same. Even when the almanac was 30-days old, the maximum difference for these sample points was less than 3%. However, there was a general trend that the differences became larger as the almanac data became more outdated. This is expected as the more outdated almanac data would be expected to have less accurate corrections on the orbital deviations of the satellites compared to more recent almanac data.

### 3. Difference in Number of Visible Satellites

However, in the case where the total number of visible satellites is different, large errors may be possible. In this case, the differences in DOP values were found to be as

high as 25% in one of the sample points when the 30-days old almanac was used. Previous sections in the report had already discussed the effect of having one more or less visible satellite on the DOP.

However, note that the reason for the difference in the total number of visible satellite is different when it comes to the use of outdated almanac data. In this case, the use of such data in the calculations resulted in satellite positions that were slightly different compared to the satellite positions if the correct almanac data were used. As a result, when a satellite is near the visibility criterion, the exact time when the satellite goes into or out of the field of view is different depending on the almanac data used. The difference in the exact time increases when the almanac is more outdated hence, the probability of getting the wrong set of satellites in the field of view increases as the almanac data becomes more outdated. Hence, wherever possible, the most recent almanac data should be used.

Details on the DOP values calculated from outdated almanac data are shown in Appendix G.

## C. AVERAGE HDOP AND VDOP

In the event that time and location cannot be specified ahead of time, it is still useful to have a "rule of thumb" on the DOP values during mission planning. The most useful DOP values for GPS-guided weapons are the HDOP and VDOP as these values are used in calculations to estimate the Circular and Height Error Probable (CEP & HEP). The current "rule of thumb" regarding the ratio between VDOP and HDOP is that the VDOP is twice the value of HDOP [13].

### 1. Approach

In order to verify the current "rule of thumb" as well as to establish another for the global average value of HDOP and VDOP, one million iterations of the DOP calculation were done to obtain the average DOP values for random positions on the Earth's surface (i.e., altitude is zero) at random time for a specified day. Six different days were sampled and the mean of the average DOP values from each day is taken to be the global average.

Three values of obstruction angle, $\beta = 0^o$, $10^o$ and $15^o$, were used to get three sets of values for HDOP and VDOP. $\beta = 0^o$ represented the most optimistic case where there is no obstruction to the field of view of the GPS receiver. This, however, may not be realistic in many situations hence, $\beta = 10^o$ and $15^o$ were used to represent cases that are more realistic.

In order to get the statistical spread of the ratio between VDOP and HDOP, the VDOP and HDOP generated by the Matlab program that were used to compare against results generated by the Trimble program in the previous sections were extracted and the ratio between the two DOP values were calculated for each sample point.

**2. Results**

The results on the global average VDOP and HDOP are shown in Table 5 below.

Table 5. Global Average VDOP and HDOP

| $\beta$ | Global Average VDOP | Global Average HDOP | Ratio between Global Average VDOP and Global Average HDOP |
|---|---|---|---|
| $0^o$ | 1.2 | 0.8 | 1.5 : 1 |
| $10^o$ | 1.8 | 1.0 | 1.8 : 1 |
| $15^o$ | 2.4 | 1.2 | 2.0 : 1 |

From the results above, the "rule of thumb" that the VDOP value is twice the value of HDOP was found to be most accurate when $\beta$ is $15^o$. The ratio between VDOP and HDOP was observed to increase as $\beta$ increase. However, note that the trend of increasing ratio would not continue for large $\beta$. This is because fewer satellites would be visible as $\beta$ increases and there would come a point where there were less than four satellites in the field of view- the minimum number to calculate the GPS receiver's position. Details on the global average of DOP are shown in Appendix H.

When the extracted VDOP and HDOP values from the 266 samples generated by the Matlab program were examined, the ratio between VDOP and HDOP for $\beta = 10^{o}$ was found to vary widely from about 0.8 to 3.6 with an average and standard deviation of about 1.8 and 0.4 respectively.

Although the "rule of thumb" that the VDOP is twice the value of HDOP remained to be a simple and reasonable guide, caution should be exercised as the spread of the ratio was shown to be fairly large. The study also gives planners the flexibility to use other ratios if more information is known about the obstructions to the field of view of the GPS receiver in the operating area. Moreover, the study provided planners with the average values of VDOP and HDOP that can be used if actual DOP values are not readily available.

Details on the results are shown in Appendix I. Note that the highest and lowest ratio between VDOP and HDOP are shaded in grey for clarity.

THIS PAGE INTENTIONALLY LEFT BLANK

# VI. CONCLUSION

A DOP calculation program that extracts satellite information from the SEM almanac file had been successfully developed in the Visual C++ environment. Comparison of the results with a commercial DOP calculation program, the Trimble Planning software, showed that the algorithm had been implemented correctly.

The effect of a difference in the number of visible satellites on DOP had been investigated. It was found the addition or loss of one satellite from the field of view has the potential to result in disproportional effects on the DOP values depending on the configuration of the rest of the visible satellites. The change in DOP values had been found to be as large as 38%.

The effects of using outdated almanac data, up to 30-days old, had also been studied. It was found that the effect on DOP is small unless the number of satellites in the field of view of the GPS receiver is different. The more outdated the almanac data, the higher the probability of encountering cases of different number of visible satellites.

It should be highlighted that, although discrepancies in number of visible satellites could be attributed to small differences in the algorithm between the Matlab and Trimble program, the level of uncertainty during actual GPS usage can be expected to have a much larger effect. Isolated buildings or hills may block the signal from an otherwise visible satellite, causing the effective number of visible satellites to decrease. Hence, an important take-away is to realize the potential uncertainty in the DOP values where conditions on the ground can deviate from the theoretical ideal calculated in the program.

Finally, the relationship between HDOP and VDOP had been studied. A "rule of thumb" on the global average values of HDOP and VDOP was established. This would allow planners to have a better estimate on weapon delivery accuracy when specific time and position of target are unknown.

## A. FUTURE WORK

### 1. Integration of Code into JWS

The next stage of development of the program is to integrate the codes for DOP calculation into the JWS. As the JWS operates in a secure network, direct access to Internet is not possible. As such, the web address to download the latest almanac file as well as the location to retrieve the stored almanac file in the secure network would need to be updated in the code.

### 2. Enhancement of GUI

Throughout the program development, the user-friendliness of the GUI had been a consideration. However, due to limited time, feedback from potential users of the program had not been solicited. Future versions of the program should elicit feedback from users to improve the usability and functionality of the program.

# APPENDIX A. EXAMPLES OF ALMANAC FILE

## A. EXAMPLE OF ALMANAC FILE IN SEM (.AL3) FORMAT

```
31  CURRENT.ALM
 493 405504

2
61
0
 8.95786285400391E-03 -3.29971313476562E-04 -2.54294718615711E-09
 5.15354736328125E+03  5.08018851280212E-01  8.63925099372864E-01
-3.86929750442505E-01  1.58309936523438E-04  0.00000000000000E+00
0
9

3
33
0
 1.16286277770996E-02 -5.10406494140625E-03 -2.58296495303512E-09
 5.15366699218750E+03  1.43044233322144E-01  2.83277988433838E-01
-5.01049160957336E-01  3.49044799804688E-04  3.63797880709171E-12
0
9

4
34
0
 8.63361358642578E-03 -5.72204589843750E-04 -2.56113708019257E-09
 5.15361083984375E+03  5.13915061950684E-01  1.50653243064880E-01
 5.15799283981323E-01 -3.26156616210938E-04 -1.45519152283669E-11
0
9
```

5
35
1
 9.39798355102539E-03  8.81195068359375E-04 -2.50292941927910E-09
 5.15357714843750E+03 -1.85006141662598E-01  4.12206888198853E-01
 3.90790939331055E-01 -6.39915466308594E-04 -4.36557456851006E-11
0
9

6
36
0
 5.70201873779297E-03 -2.72560119628906E-03 -2.56113708019257E-09
 5.15364599609375E+03  1.64265394210815E-01 -4.65690255165100E-01
 3.02691578865051E-01  6.77108764648438E-05 -1.45519152283669E-11
0
9

7
48
0
 2.29692459106445E-03  7.37380981445312E-03 -2.57568899542093E-09
 5.15363134765625E+03 -4.88172769546509E-01  9.30595278739929E-01
-9.65989708900452E-01  2.28881835937500E-05  0.00000000000000E+00
0
10

8
38
0
 1.04718208312988E-02  1.41143798828125E-02 -2.49929144047201E-09
 5.15364062500000E+03 -4.69682693481445E-01  9.47455763816833E-01
 8.20714473724365E-01 -1.89781188964844E-04  0.00000000000000E+00
0
9

9
39
0
 2.01759338378906E-02  1.02252960205078E-02 -2.55022314377129E-09
 5.15360449218750E+03 -4.99263167381287E-01  4.64545011520386E-01
 7.39052772521973E-01  3.91006469726562E-05  3.63797880709171E-12
0
9

10
40
0
 8.52298736572266E-03  5.58662414550781E-03 -2.46654963120818E-09
 5.15373876953125E+03  8.53048920631409E-01  1.88068747520447E-01
-1.12927794456482E-01 -1.04904174804688E-05  0.00000000000000E+00
0
9

11
46
0
 9.18340682983398E-03 -1.68552398681641E-02 -2.74667399935424E-09
 5.15358886718750E+03  4.46067214012146E-01  2.17745780944824E-01
 9.90043640136719E-01  8.58306884765625E-06  0.00000000000000E+00
0
9

12
58
0
 3.19671630859375E-03  7.56835937500000E-03 -2.42289388552308E-09
 5.15368066406250E+03 -1.53432011604309E-01 -2.58711695671082E-01
-9.05213356018066E-01 -3.18527221679688E-04  3.63797880709171E-12
0
10

13
43
0
 3.74412536621094E-03  1.68495178222656E-02 -2.49201548285782E-09
 5.15371533203125E+03 -8.04313659667969E-01  4.79433178901672E-01
-1.70316457748413E-01  2.88009643554688E-04  0.00000000000000E+00
0
9

14
41
0
 4.41741943359375E-03  1.48086547851562E-02 -2.51020537689328E-09
 5.15352929687500E+03 -8.10300111770630E-01 -6.68291091918945E-01
-2.97486662864685E-01 -1.58309936523438E-04  3.63797880709171E-12
0
9

15
55
0
 1.40905380249023E-03  4.67300415039062E-03 -2.60115484707057E-09
 5.15352832031250E+03 -8.25886249542236E-01 -2.36185193061829E-01
-1.36659026145935E-01 -2.52723693847656E-04 -3.63797880709171E-12
0
10

16
56
0
 5.04970550537109E-03  8.05664062500000E-03 -2.42653186433017E-09
 5.15371533203125E+03 -1.48093223571777E-01 -1.22860670089722E-01
 2.92908310890198E-01  7.72476196289062E-05 -3.63797880709171E-12
0
9

17
53
0
 4.19712066650391E-03  5.53703308105469E-03 -2.48110154643655E-09
 5.15357861328125E+03  1.83424353599548E-01 -8.75360608100891E-01
-5.27738332748413E-02  4.57763671875000E-05  0.00000000000000E+00
0
10

18
54
0
 9.91773605346680E-03  5.11169433593750E-04 -2.51020537689328E-09
 5.15370654296875E+03  8.55316162109375E-01 -7.76121497154236E-01
 3.33639025688171E-01 -8.58306884765625E-05  3.63797880709171E-12
0
9

19
59
0
 5.12599945068359E-03  4.80270385742188E-03 -2.49565346166492E-09
 5.15372167968750E+03  2.01364517211914E-01 -1.24549746513367E-01
-2.88856863975525E-01  3.24249267578125E-05  0.00000000000000E+00
0
9

20
51
0
 3.94058227539062E-03  3.71932983398438E-04 -2.51748133450747E-09
 5.15363427734375E+03  8.38206768035889E-01  4.18452620506287E-01
 3.80490541458130E-01  9.15527343750000E-05  0.00000000000000E+00
0
9

21
45
0
 1.46293640136719E-02 -2.63404846191406E-03 -2.57568899542093E-09
 5.15357958984375E+03  5.18336772918701E-01 -8.43170762062073E-01
 7.39881634712219E-01  3.05175781250000E-05  0.00000000000000E+00
0
9

22
47
1
 4.88758087158203E-03 -2.32696533203125E-04 -2.51384335570037E-09
 5.15361767578125E+03  8.57203364372253E-01 -5.80405235290527E-01
-2.43811607360840E-02  2.05993652343750E-04  0.00000000000000E+00
0
9

23
60
0
 5.77783584594727E-03  9.09423828125000E-03 -2.56477505899966E-09
 5.15360058593750E+03 -8.20027351379395E-01  9.17497277259827E-01
-4.60470557212830E-01  3.86238098144531E-04  0.00000000000000E+00
0
9

24
24
0
 6.95610046386719E-03  2.66265869140625E-03 -2.52475729212165E-09
 5.15360839843750E+03  5.27971267700195E-01 -2.03566551208496E-01
 3.01274657249451E-01  1.63078308105469E-04  3.63797880709171E-12
0
9

25
25
1
 1.17974281311035E-02  8.11958312988281E-03 -2.57568899542093E-09
 5.15362548828125E+03 -5.18336057662964E-01 -3.91333818435669E-01
 4.67219591140747E-01  2.23159790039062E-04  2.54658516496420E-11
0
9

26
26
0
 1.95465087890625E-02  1.61743164062500E-02 -2.49201548285782E-09
 5.15360107421875E+03 -8.04983854293823E-01  3.08019638061523E-01
-6.00259423255920E-01  4.19616699218750E-05  3.63797880709171E-12
0
9

27
27
1
 2.10742950439453E-02  9.59968566894531E-03 -2.53567122854292E-09
 5.15513378906250E+03 -5.06914854049683E-01 -5.25584697723389E-01
 1.71768665313721E-01  2.19345092773438E-05  3.63797880709171E-12
0
9

28
44
1
 1.46369934082031E-02  7.48062133789062E-03 -2.42289388552308E-09
 5.15359619140625E+03 -1.45439743995667E-01 -6.54806971549988E-01
 8.22650194168091E-02 -2.38418579101562E-05  0.00000000000000E+00
0
9

29
57
0
 3.26204299926758E-03  5.59425354003906E-03 -2.48837750405073E-09
 5.15368408203125E+03  1.86028838157654E-01 -4.27628159523010E-01
 7.56655812263489E-01 -1.23977661132812E-05  3.63797880709171E-12
0
10

30
30
0
 1.11746788024902E-02  2.77900695800781E-03 -2.48110154643655E-09
 5.15359765625000E+03 -1.68058753013611E-01  4.49827551841736E-01
 2.19786643981934E-01  1.24931335449219E-04  3.63797880709171E-12
0
9

31
52
1
 7.18879699707031E-03  9.09423828125000E-03 -2.55386112257838E-09
 5.15359130859375E+03 -4.89016652107239E-01 -3.80007505416870E-01
-9.89319086074829E-01 -5.05447387695312E-05  0.00000000000000E+00
0
10

32
23
0
 1.35531425476074E-02  7.39669799804688E-03 -2.44108377955854E-09
 5.15360058593750E+03  8.72316479682922E-01 -3.82601380348206E-01
-7.17132568359375E-01  2.97546386718750E-04 -3.63797880709171E-12
0
9

## B. EXAMPLE OF ALMANAC FILE IN YUMA (.ALM) FORMAT

******** Week 493 almanac for PRN-02 ********
ID: 02
Health: 000
Eccentricity: 0.8957862854E-002
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9414405823
Rate of Right Ascen(r/s): -0.7992639439E-008
SQRT(A) (m 1/2): 5153.547363
Right Ascen at Week(rad): 0.1595988274E+001
Argument of Perigee(rad): 2.714100718
Mean Anom(rad): -0.1215575695E+001
Af0(s): 0.1583099365E-003
Af1(s/s): 0.0000000000E+000
week: 493

******** Week 493 almanac for PRN-03 ********
ID: 03
Health: 000
Eccentricity: 0.1162862778E-001
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9264431000
Rate of Right Ascen(r/s): -0.8116330719E-008
SQRT(A) (m 1/2): 5153.666992
Right Ascen at Week(rad): 0.4493867159E+000
Argument of Perigee(rad): 0.889944077
Mean Anom(rad): -0.1574092388E+001
Af0(s): 0.3490447998E-003
Af1(s/s): 0.3637978807E-011
week: 493

******** Week 493 almanac for PRN-04 ********
ID: 04
Health: 000
Eccentricity: 0.8633613586E-002
Time of Applicability(s): 405504.0000

Orbital Inclination(rad): 0.9406795502
Rate of Right Ascen(r/s): -0.8043571142E-008
SQRT(A) (m 1/2): 5153.610840
Right Ascen at Week(rad): 0.1614511728E+001
Argument of Perigee(rad): 0.473291159
Mean Anom(rad): 0.1620431185E+001
Af0(s): -0.3261566162E-003
Af1(s/s): -0.1455191523E-010
week: 493

******** Week 493 almanac for PRN-05 ********
ID: 05
Health: 000
Eccentricity: 0.9397983551E-002
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9452457428
Rate of Right Ascen(r/s): -0.7858034223E-008
SQRT(A) (m 1/2): 5153.577148
Right Ascen at Week(rad): -0.5812139511E+000
Argument of Perigee(rad): 1.294986129
Mean Anom(rad): 0.1227705956E+001
Af0(s): -0.6399154663E-003
Af1(s/s): -0.4365574569E-010
week: 493

******** Week 493 almanac for PRN-06 ********
ID: 06
Health: 000
Eccentricity: 0.5702018738E-002
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9339141846
Rate of Right Ascen(r/s): -0.8050847100E-008
SQRT(A) (m 1/2): 5153.645996
Right Ascen at Week(rad): 0.5160549879E+000
Argument of Perigee(rad): -1.463009119
Mean Anom(rad): 0.9509336948E+000

Af0(s): 0.6771087646E-004
Af1(s/s): -0.1455191523E-010
week: 493

******** Week 493 almanac for PRN-07 ********
ID: 07
Health: 000
Eccentricity: 0.2296924591E-002
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9656429291
Rate of Right Ascen(r/s): -0.8090864867E-008
SQRT(A) (m 1/2): 5153.631348
Right Ascen at Week(rad): -0.1533640027E+001
Argument of Perigee(rad): 2.923551321
Mean Anom(rad): -0.3034746170E+001
Af0(s): 0.2288818359E-004
Af1(s/s): 0.0000000000E+000
week: 493

******** Week 493 almanac for PRN-08 ********
ID: 08
Health: 000
Eccentricity: 0.1047182083E-001
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9868202209
Rate of Right Ascen(r/s): -0.7847120287E-008
SQRT(A) (m 1/2): 5153.640625
Right Ascen at Week(rad): -0.1475551724E+001
Argument of Perigee(rad): 2.976520061
Mean Anom(rad): 0.2578350544E+001
Af0(s): -0.1897811890E-003
Af1(s/s): 0.0000000000E+000
week: 493

******** Week 493 almanac for PRN-09 ********
ID: 09

Health: 000
Eccentricity: 0.2017593384E-001
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9746017456
Rate of Right Ascen(r/s): -0.8014467312E-008
SQRT(A) (m 1/2): 5153.604492
Right Ascen at Week(rad): -0.1568481445E+001
Argument of Perigee(rad): 1.459411144
Mean Anom(rad): 0.2321802735E+001
Af0(s): 0.3910064697E-004
Af1(s/s): 0.3637978807E-011
week: 493

******** Week 493 almanac for PRN-10 ********
ID: 10
Health: 000
Eccentricity: 0.8522987366E-002
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9600296021
Rate of Right Ascen(r/s): -0.7745256880E-008
SQRT(A) (m 1/2): 5153.738770
Right Ascen at Week(rad): 0.2679932237E+001
Argument of Perigee(rad): 0.590835452
Mean Anom(rad): -0.3547731638E+000
Af0(s): -0.1049041748E-004
Af1(s/s): 0.0000000000E+000
week: 493

******** Week 493 almanac for PRN-11 ********
ID: 11
Health: 000
Eccentricity: 0.9183406830E-002
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.8895263672
Rate of Right Ascen(r/s): -0.8625647752E-008
SQRT(A) (m 1/2): 5153.588867

Right Ascen at Week(rad): 0.1401361465E+001
Argument of Perigee(rad): 0.684068561
Mean Anom(rad): 0.3110313773E+001
Af0(s): 0.8583068848E-005
Af1(s/s): 0.0000000000E+000
week: 493

******** Week 493 almanac for PRN-12 ********
ID: 12
Health: 000
Eccentricity: 0.3196716309E-002
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9662551880
Rate of Right Ascen(r/s): -0.7617927622E-008
SQRT(A) (m 1/2): 5153.680664
Right Ascen at Week(rad): -0.4820208549E+000
Argument of Perigee(rad): -0.812766790
Mean Anom(rad): -0.2843811631E+001
Af0(s): -0.3185272217E-003
Af1(s/s): 0.3637978807E-011
week: 493

******** Week 493 almanac for PRN-13 ********
ID: 13
Health: 000
Eccentricity: 0.3744125366E-002
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9954128265
Rate of Right Ascen(r/s): -0.7832568372E-008
SQRT(A) (m 1/2): 5153.715332
Right Ascen at Week(rad): -0.2526825905E+001
Argument of Perigee(rad): 1.506183743
Mean Anom(rad): -0.5350649357E+000
Af0(s): 0.2880096436E-003
Af1(s/s): 0.0000000000E+000
week: 493

******** Week 493 almanac for PRN-14 ********
ID: 14
Health: 000
Eccentricity: 0.4417419434E-002
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9890003204
Rate of Right Ascen(r/s): -0.7883500075E-008
SQRT(A) (m 1/2): 5153.529297
Right Ascen at Week(rad): -0.2545632839E+001
Argument of Perigee(rad): -2.099498391
Mean Anom(rad): -0.9345818758E+000
Af0(s): -0.1583099365E-003
Af1(s/s): 0.3637978807E-011
week: 493

******** Week 493 almanac for PRN-15 ********
ID: 15
Health: 000
Eccentricity: 0.1409053802E-002
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9571590424
Rate of Right Ascen(r/s): -0.8178176358E-008
SQRT(A) (m 1/2): 5153.528320
Right Ascen at Week(rad): -0.2594598174E+001
Argument of Perigee(rad): -0.741997719
Mean Anom(rad): -0.4293270111E+000
Af0(s): -0.2527236938E-003
Af1(s/s): -0.3637978807E-011
week: 493

******** Week 493 almanac for PRN-16 ********
ID: 16
Health: 000
Eccentricity: 0.5049705505E-002
Time of Applicability(s): 405504.0000

Orbital Inclination(rad): 0.9677886963
Rate of Right Ascen(r/s): -0.7625203580E-008
SQRT(A) (m 1/2): 5153.715332
Right Ascen at Week(rad): -0.4652485847E+000
Argument of Perigee(rad): -0.385978222
Mean Anom(rad): 0.9201985598E+000
Af0(s): 0.7724761963E-004
Af1(s/s): -0.3637978807E-011
week: 493

******** Week 493 almanac for PRN-17 ********
ID: 17
Health: 000
Eccentricity: 0.4197120667E-002
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9598731995
Rate of Right Ascen(r/s): -0.7799826562E-008
SQRT(A) (m 1/2): 5153.578613
Right Ascen at Week(rad): 0.5762445927E+000
Argument of Perigee(rad): -2.750026464
Mean Anom(rad): -0.1657938957E+000
Af0(s): 0.4577636719E-004
Af1(s/s): 0.0000000000E+000
week: 493

******** Week 493 almanac for PRN-18 ********
ID: 18
Health: 000
Eccentricity: 0.9917736053E-002
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9440841675
Rate of Right Ascen(r/s): -0.7879862096E-008
SQRT(A) (m 1/2): 5153.706543
Right Ascen at Week(rad): 0.2687054992E+001
Argument of Perigee(rad): -2.438257575
Mean Anom(rad): 0.1048157930E+001

Af0(s): -0.8583068848E-004
Af1(s/s): 0.3637978807E-011
week: 493

******** Week 493 almanac for PRN-19 ********
ID: 19
Health: 000
Eccentricity: 0.5125999451E-002
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9575653076
Rate of Right Ascen(r/s): -0.7843482308E-008
SQRT(A) (m 1/2): 5153.721680
Right Ascen at Week(rad): 0.6326053143E+000
Argument of Perigee(rad): -0.391284585
Mean Anom(rad): -0.9074705839E+000
Af0(s): 0.3242492676E-004
Af1(s/s): 0.0000000000E+000
week: 493

******** Week 493 almanac for PRN-20 ********
ID: 20
Health: 000
Eccentricity: 0.3940582275E-002
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9436454773
Rate of Right Ascen(r/s): -0.7901689969E-008
SQRT(A) (m 1/2): 5153.634277
Right Ascen at Week(rad): 0.2633304238E+001
Argument of Perigee(rad): 1.314607620
Mean Anom(rad): 0.1195346236E+001
Af0(s): 0.9155273438E-004
Af1(s/s): 0.0000000000E+000
week: 493

******** Week 493 almanac for PRN-21 ********
ID: 21

Health: 000
Eccentricity: 0.1462936401E-001
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9342021942
Rate of Right Ascen(r/s): -0.8087226888E-008
SQRT(A) (m 1/2): 5153.579590
Right Ascen at Week(rad): 0.1628402948E+001
Argument of Perigee(rad): -2.648899078
Mean Anom(rad): 0.2324406743E+001
Af0(s): 0.3051757812E-004
Af1(s/s): 0.0000000000E+000
week: 493

******** Week 493 almanac for PRN-22 ********
ID: 22
Health: 000
Eccentricity: 0.4887580872E-002
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9417476654
Rate of Right Ascen(r/s): -0.7898051990E-008
SQRT(A) (m 1/2): 5153.617676
Right Ascen at Week(rad): 0.2692983747E+001
Argument of Perigee(rad): -1.823396802
Mean Anom(rad): -0.7659566402E-001
Af0(s): 0.2059936523E-003
Af1(s/s): 0.0000000000E+000
week: 493

******** Week 493 almanac for PRN-23 ********
ID: 23
Health: 000
Eccentricity: 0.5777835846E-002
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9710483551
Rate of Right Ascen(r/s): -0.8054485079E-008
SQRT(A) (m 1/2): 5153.600586

Right Ascen at Week(rad): -0.2576191902E+001
Argument of Perigee(rad): 2.882402658
Mean Anom(rad): -0.1446610928E+001
Af0(s): 0.3862380981E-003
Af1(s/s): 0.0000000000E+000
week: 493

******** Week 493 almanac for PRN-24 ********
ID: 24
Health: 000
Eccentricity: 0.6956100464E-002
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9508419037
Rate of Right Ascen(r/s): -0.7930793799E-008
SQRT(A) (m 1/2): 5153.608398
Right Ascen at Week(rad): 0.1658670664E+001
Argument of Perigee(rad): -0.639523149
Mean Anom(rad): 0.9464823008E+000
Af0(s): 0.1630783081E-003
Af1(s/s): 0.3637978807E-011
week: 493

******** Week 493 almanac for PRN-25 ********
ID: 25
Health: 000
Eccentricity: 0.1179742813E-001
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9679870605
Rate of Right Ascen(r/s): -0.8098140825E-008
SQRT(A) (m 1/2): 5153.625488
Right Ascen at Week(rad): -0.1628400803E+001
Argument of Perigee(rad): -1.229411483
Mean Anom(rad): 0.1467813611E+001
Af0(s): 0.2231597900E-003
Af1(s/s): 0.2546585165E-010
week: 493

******** Week 493 almanac for PRN-26 ********
ID: 26
Health: 000
Eccentricity: 0.1954650879E-001
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9932918549
Rate of Right Ascen(r/s): -0.7832568372E-008
SQRT(A) (m 1/2): 5153.601074
Right Ascen at Week(rad): -0.2528931379E+001
Argument of Perigee(rad): 0.967672229
Mean Anom(rad): -0.1885770559E+001
Af0(s): 0.4196166992E-004
Af1(s/s): 0.3637978807E-011
week: 493

******** Week 493 almanac for PRN-27 ********
ID: 27
Health: 000
Eccentricity: 0.2107429504E-001
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9726352692
Rate of Right Ascen(r/s): -0.7970811566E-008
SQRT(A) (m 1/2): 5155.133789
Right Ascen at Week(rad): -0.1592519999E+001
Argument of Perigee(rad): -1.651172996
Mean Anom(rad): 0.5396271944E+000
Af0(s): 0.2193450928E-004
Af1(s/s): 0.3637978807E-011
week: 493

******** Week 493 almanac for PRN-28 ********
ID: 28
Health: 000
Eccentricity: 0.1463699341E-001
Time of Applicability(s): 405504.0000

Orbital Inclination(rad): 0.9659786224
Rate of Right Ascen(r/s): -0.7614289643E-008
SQRT(A) (m 1/2): 5153.596191
Right Ascen at Week(rad): -0.4569123983E+000
Argument of Perigee(rad): -2.057136774
Mean Anom(rad): 0.2584432364E+000
Af0(s): -0.2384185791E-004
Af1(s/s): 0.0000000000E+000
week: 493

******** Week 493 almanac for PRN-29 ********
ID: 29
Health: 000
Eccentricity: 0.3262042999E-002
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9600524902
Rate of Right Ascen(r/s): -0.7814378478E-008
SQRT(A) (m 1/2): 5153.684082
Right Ascen at Week(rad): 0.5844268799E+000
Argument of Perigee(rad): -1.343433499
Mean Anom(rad): 0.2377104282E+001
Af0(s): -0.1239776611E-004
Af1(s/s): 0.3637978807E-011
week: 493

******** Week 493 almanac for PRN-30 ********
ID: 30
Health: 000
Eccentricity: 0.1117467880E-001
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9512081146
Rate of Right Ascen(r/s): -0.7788912626E-008
SQRT(A) (m 1/2): 5153.597656
Right Ascen at Week(rad): -0.5279721022E+000
Argument of Perigee(rad): 1.413174987
Mean Anom(rad): 0.6904801130E+000

Af0(s): 0.1249313354E-003
Af1(s/s): 0.3637978807E-011
week: 493

******** Week 493 almanac for PRN-31 ********
ID: 31
Health: 000
Eccentricity: 0.7188796997E-002
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9710483551
Rate of Right Ascen(r/s): -0.8021743270E-008
SQRT(A) (m 1/2): 5153.591309
Right Ascen at Week(rad): -0.1536291122E+001
Argument of Perigee(rad): -1.193828821
Mean Anom(rad): -0.3108037591E+001
Af0(s): -0.5054473877E-004
Af1(s/s): 0.0000000000E+000
week: 493

******** Week 493 almanac for PRN-32 ********
ID: 32
Health: 000
Eccentricity: 0.1355314255E-001
Time of Applicability(s): 405504.0000
Orbital Inclination(rad): 0.9657154083
Rate of Right Ascen(r/s): -0.7672497304E-008
SQRT(A) (m 1/2): 5153.600586
Right Ascen at Week(rad): 0.2740463018E+001
Argument of Perigee(rad): -1.201977730
Mean Anom(rad): -0.2252938390E+001
Af0(s): 0.2975463867E-003
Af1(s/s): -0.3637978807E-011
week: 493

THIS PAGE INTENTIONALLY LEFT BLANK

# APPENDIX B. VISUAL C++ CODES FOR DOP CALCULATION

## A. DOP_CALCULATORDLG.CPP

```
// DOP_CalculatorDlg.cpp : implementation file

#include <tchar.h>
#include <urlmon.h>
#pragma comment(lib, "urlmon.lib")
#include "stdafx.h"
#include <math.h>
#include <stdlib.h>
#include <string>
#include <iostream>
#include <fstream>
#include "DOP_Calculator.h"
#include "DOP_CalculatorDlg.h"
#include "Matrix.h"

using namespace std;

#ifdef _DEBUG
#define new DEBUG_NEW
#endif

// CAboutDlg dialog used for App About

class CAboutDlg : public CDialog
{
public:
	CAboutDlg();

// Dialog Data
	enum { IDD = IDD_ABOUTBOX };

	protected:
```

```
	virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support

// Implementation
protected:
	DECLARE_MESSAGE_MAP()
};

CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
}

void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
}

BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
END_MESSAGE_MAP()

// CDOP_CalculatorDlg dialog

CDOP_CalculatorDlg::CDOP_CalculatorDlg(CWnd* pParent /*=NULL*/)
	: CDialog(CDOP_CalculatorDlg::IDD, pParent)
	, yr(2009)
	, mth(2)
	, day(3)
	, hr(0)
	, mi(0)
	, timezone(0)
	, longi(0)
	, lat(0)
	, alt(0)
	, obs(10)
	, xdop0(_T(""))
	, ydop0(_T(""))
	, vdop0(_T(""))
```

```
	, tdop0(_T(""))
	, hdop0(_T(""))
	, pdop0(_T(""))
	, gdop0(_T(""))
	, daylightsaving(FALSE)
{
	m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);

	m_pGraph=NULL;
}

void CDOP_CalculatorDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);

	DDX_Text(pDX, IDC_YR, yr);
	DDX_Text(pDX, IDC_MTH, mth);
	DDV_MinMaxDouble(pDX, mth, 1, 12);
	DDX_Text(pDX, IDC_DAY, day);
	DDV_MinMaxDouble(pDX, day, 1, 31);
	DDX_Text(pDX, IDC_HR, hr);
	DDV_MinMaxDouble(pDX, hr, 0, 23);
	DDX_Text(pDX, IDC_MI, mi);
	DDV_MinMaxDouble(pDX, mi, 0, 59);
	DDX_Text(pDX, IDC_TIMEZONE, timezone);
	DDX_Text(pDX, IDC_LONGI, longi);
	DDV_MinMaxDouble(pDX, longi, -180, 180);
	DDX_Text(pDX, IDC_LAT, lat);
	DDV_MinMaxDouble(pDX, lat, -90, 90);
	DDX_Text(pDX, IDC_ALT, alt);
	DDX_Text(pDX, IDC_OBS, obs);
	DDX_Text(pDX, IDC_XDOP0, xdop0);
	DDX_Text(pDX, IDC_YDOP0, ydop0);
	DDX_Text(pDX, IDC_VDOP0, vdop0);
	DDX_Text(pDX, IDC_TDOP0, tdop0);
	DDX_Text(pDX, IDC_HDOP0, hdop0);
```

```
	DDX_Text(pDX, IDC_PDOP0, pdop0);
	DDX_Text(pDX, IDC_GDOP0, gdop0);
	DDX_Check(pDX, IDC_DAYLIGHTSAV, daylightsaving);
}

BEGIN_MESSAGE_MAP(CDOP_CalculatorDlg, CDialog)
	ON_WM_SYSCOMMAND()
	ON_WM_PAINT()
	ON_WM_QUERYDRAGICON()
	//}}AFX_MSG_MAP
	ON_BN_CLICKED(IDC_ALMANAC, &CDOP_CalculatorDlg::OnBnClickedAlmanac)
	ON_BN_CLICKED(IDC_COMPUTE, &CDOP_CalculatorDlg::OnBnClickedCompute)
	ON_BN_CLICKED(IDC_XDOP_RADIO, &CDOP_CalculatorDlg::OnBnClickedXdopRadio)
	ON_BN_CLICKED(IDC_YDOP_RADIO, &CDOP_CalculatorDlg::OnBnClickedYdopRadio)
	ON_BN_CLICKED(IDC_VDOP_RADIO, &CDOP_CalculatorDlg::OnBnClickedVdopRadio)
	ON_BN_CLICKED(IDC_TDOP_RADIO, &CDOP_CalculatorDlg::OnBnClickedTdopRadio)
	ON_BN_CLICKED(IDC_HDOP_RADIO, &CDOP_CalculatorDlg::OnBnClickedHdopRadio)
	ON_BN_CLICKED(IDC_PDOP_RADIO, &CDOP_CalculatorDlg::OnBnClickedPdopRadio)
	ON_BN_CLICKED(IDC_GDOP_RADIO, &CDOP_CalculatorDlg::OnBnClickedGdopRadio)
END_MESSAGE_MAP()

// CDOP_CalculatorDlg message handlers

BOOL CDOP_CalculatorDlg::OnInitDialog()
{
	CDialog::OnInitDialog();

	// Add "About..." menu item to system menu.

	// IDM_ABOUTBOX must be in the system command range.
	ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
	ASSERT(IDM_ABOUTBOX < 0xF000);

	CMenu* pSysMenu = GetSystemMenu(FALSE);
	if (pSysMenu != NULL)
	{
```

```
		CString strAboutMenu;
		strAboutMenu.LoadString(IDS_ABOUTBOX);
		if (!strAboutMenu.IsEmpty())
		{
			pSysMenu->AppendMenu(MF_SEPARATOR);
			pSysMenu->AppendMenu(MF_STRING, IDM_ABOUTBOX, strAboutMenu);
		}
	}

	// Set the icon for this dialog.  The framework does this automatically
	//  when the application's main window is not a dialog
	SetIcon(m_hIcon, TRUE);			// Set big icon
	SetIcon(m_hIcon, FALSE);		// Set small icon

	// TODO: Add extra initialization here
	//This change the position of default graph
	m_pGraph=new CGraph(this,230,230,300,250,G_WHITESCHEME);
	m_lpfs=NULL;
	m_pPlotItems=NULL;

	almanac_location = "H:\\MyDocs\\Visual Studio 2005\\Projects\\Test3\\Test3\\current.al3";
	got_data = false;

	return TRUE;  // return TRUE  unless you set the focus to a control
}

void CDOP_CalculatorDlg::OnSysCommand(UINT nID, LPARAM lParam)
{
	if ((nID & 0xFFF0) == IDM_ABOUTBOX)
	{
		CAboutDlg dlgAbout;
		dlgAbout.DoModal();
	}
	else
	{
		CDialog::OnSysCommand(nID, lParam);
```

```
	}
}

// If you add a minimize button to your dialog, you will need the code below
//  to draw the icon.  For MFC applications using the document/view model,
//  this is automatically done for you by the framework.

void CDOP_CalculatorDlg::OnPaint()
{
	if (IsIconic())
	{
		CPaintDC dc(this); // device context for painting

		SendMessage(WM_ICONERASEBKGND,
reinterpret_cast<WPARAM>(dc.GetSafeHdc()), 0);

		// Center icon in client rectangle
		int cxIcon = GetSystemMetrics(SM_CXICON);
		int cyIcon = GetSystemMetrics(SM_CYICON);
		CRect rect;
		GetClientRect(&rect);
		int x = (rect.Width() - cxIcon + 1) / 2;
		int y = (rect.Height() - cyIcon + 1) / 2;

		// Draw the icon
		dc.DrawIcon(x, y, m_hIcon);
	}
	else
	{
		CDialog::OnPaint();

		m_pGraph->PaintGraph();
	}
}

// The system calls this function to obtain the cursor to display while the user drags
```

```
//  the minimized window.
HCURSOR CDOP_CalculatorDlg::OnQueryDragIcon()
{
	return static_cast<HCURSOR>(m_hIcon);
}



void CDOP_CalculatorDlg::OnBnClickedAlmanac()
{
	//Downloads latest almanac data from the USCG website to specified location
	if(URLDownloadToFile(NULL, _T("http://www.navcen.uscg.gov/gps/current/current.al3"),
almanac_location, 0, NULL ))
	{
		AfxMessageBox((CString)"Error: File Not Found or you are
offline",MB_ICONERROR);
	}
	else
	{
		AfxMessageBox((CString)"Latest Almanac Downloaded",MB_OK);
	}
}



void CDOP_CalculatorDlg::OnBnClickedCompute()
{
	//Clears any existing plot data
	m_lpfs=NULL;
	m_pPlotItems=NULL;

	//Updates user input
	UpdateData(TRUE);

	if (daylightsaving == true)
		daylightsav = 1;
	else
		daylightsav = 0;
```

```
//---------------------------------------------------------------------
//Convert Target's Latitude , Longitude and Altitude to ECEF Coordinates
//---------------------------------------------------------------------
//WGS84 ellopsoid constants
double a = 6378137;
double es = 8.1819190842622e-2;
double sealevel = 0; //Altitude of sealevel is zero meter

double pi = 3.1415926535898;
double lat_rad = lat * pi/180;
double long_rad = longi * pi/180;

double N = a/sqrt( 1 - ( pow(es, 2) ) * ( pow( sin(lat_rad), 2 ) ) ); //Prime vertical radius of curvature
double xTgt = (N + alt) * cos(lat_rad) * cos(long_rad); //Target x ECEF coordinates (meters)
double yTgt = (N + alt) * cos(lat_rad) * sin(long_rad); //Target y ECEF coordinates (meters)
double zTgt = ( ( (1 - pow(es, 2) ) * N) + alt ) * sin(lat_rad); //Target z ECEF coordinates (meters)

double xLocalRef = (N + sealevel) * cos(lat_rad) * cos(long_rad); //x ECEF coordinates for Local Reference Pt for ENU frame (meters)
double yLocalRef = (N + sealevel) * cos(lat_rad) * sin(long_rad); //y ECEF coordinates for Local Reference Pt for ENU frame (meters)
double zLocalRef = ( ( (1 - pow(es, 2) ) * N) + sealevel ) * sin(lat_rad); //z ECEF coordinates for Local Reference Pt for ENU frame (meters)
//---------------------------------------------------------------------
//Convert Almanac Data to Satellite Position in East-North-Up (ENU) Coordinates and Check the Line of Sight
//---------------------------------------------------------------------

//Constants
double io = 0.3 * pi; //Inclination angle @ ref. time (rad)
double mju = 3.986005e14; //WGS84 value of the Earth's universal gravitational parameter for GPS user (meter^3/sec^2)
double OMEGAdote = 7.2921151467e-5; //WGS84 value of the Earth's rotation rate (rad/sec)
double c = 2.99792458e8; //Speed of light (meter/sec)

//Reads almanac data from Almanac file
char inputline[100];
```

```
        inputline[0]='\0';

        int NumSV; //Number of satellites
        int wn; //GPS week no.
        int toa; //Time of applicability of Almanac (sec) (range: 0 to 604,784)

        //Reads the almanac data from specified location
        ifstream in(almanac_location, ios::in); //opens the almanac file
        in >> NumSV >> inputline >> wn >> toa; //reads time info from almanac file

        //Convert date to GPS time
        double JAN61980 = 44244 + 2400000.5; //Date when GPS started
        double sec_per_day = 86400;
        double JD = 367*yr - floor(7*(yr + floor((mth + 9)/12))/4) - floor(3*(floor((yr + (mth-9)/7)/100) +
1)/4) + floor(275*mth/9) + day + 1721028.5;

        double fJD = (mi/1440) + (hr/24);
        JD = JD + fJD - (timezone + daylightsav)/24;

        //Correct for leap seconds up till Mar 2009
        int GpsLeapSec;

        if    ((JD>=2444786.5)&&(JD<2445151.5))
                GpsLeapSec=1;
        else if ((JD>=2445151.5)&&(JD<2445516.5))
                GpsLeapSec=2;
        else if ((JD>=2445516.5)&&(JD<2446247.5))
                GpsLeapSec=3;
        else if ((JD>=2446247.5)&&(JD<2447161.5))
    GpsLeapSec=4;
        else if ((JD>=2447161.5)&&(JD<2447892.5))
    GpsLeapSec=5;
        else if ((JD>=2447892.5)&&(JD<2448257.5))
    GpsLeapSec=6;
        else if ((JD>=2448257.5)&&(JD<2448804.5))
    GpsLeapSec=7;
```

```
        else if ((JD>=2448804.5)&&(JD<2449169.5))
    GpsLeapSec=8;
        else if ((JD>=2449169.5)&&(JD<2449534.5))
    GpsLeapSec=9;
        else if ((JD>=2449534.5)&&(JD<2450083.5))
    GpsLeapSec=10;
        else if ((JD>=2450083.5)&&(JD<2450630.5))
    GpsLeapSec=11;
        else if ((JD>=2450630.5)&&(JD<2451179.5))
    GpsLeapSec=12;
        else if ((JD>=2451179.5)&&(JD<2453371.5))
    GpsLeapSec=13;
        else if ((JD>=2453371.5)&&(JD<2454466.5))
    GpsLeapSec=14;
        else GpsLeapSec=15; //Add when new seconds are added

        JD = JD + GpsLeapSec/sec_per_day;
        double gps_week = floor( (JD - JAN61980)/7 );
        double sec_of_week = ( (JD - JAN61980) - gps_week * 7 ) * sec_per_day;
        //End of GPS time conversion

        while (gps_week >= 1024)
        {
                gps_week = gps_week - 1024;
        }

        double tk = (gps_week - wn) * 604800 + (sec_of_week - toa); //Time since toa (sec) (range: -
302400 to 302400)

        //Need to increase the array size if there are more than 31 GPS satellites
        int PRN;
        int SVN;
        int URA;
        double ec;
        double del_ik;
        double OMEGAdot;
```

```
double sqrtA;
double OMEGAo;
double omega;
double Mo;
double Af0;
double Af1;
double SVHEALTH;
double SVCONFIG;
double EAST[31], NORTH[31], UP[31];
int LOS[31];

int i1;
for (i1=0; i1<NumSV; i1++) //reads data for different satellites
{
        //Read the data from Almanac file
        in >> PRN >> SVN >> URA;
        in >> ec >> del_ik >> OMEGAdot;
        in >> sqrtA >> OMEGAo >> omega;
        in >> Mo >> Af0 >> Af1;
        in >> SVHEALTH >> SVCONFIG;

        //Conversion of some almanac data to radians
        del_ik = del_ik * pi;
        OMEGAdot = OMEGAdot * pi;
        OMEGAo = OMEGAo * pi;
        omega = omega * pi;
        Mo = Mo * pi;

        //Position of Satellite Calculations
        double A = pow(sqrtA, 2);
        double n = sqrt(mju / pow(A, 3));
        double Mk = Mo + (tk * n); //Mean anomaly (rad)

        //Start values for iterative solution of Kepler Equation
        double Ek = Mk;
        double Eold = 0;
```

```
while ( abs(Ek - Eold) >= 1e-10 )
{
	Eold = Ek;
	Ek = Mk + ec*sin(Ek);
}

double vk = atan2( ( sqrt(1 - pow(ec, 2) ) * sin(Ek) )/( 1 - ec * cos(Ek) ), ( cos(Ek) - ec )/( 1 - ec * cos(Ek) ) ); //True Anomaly (rad)
Ek = acos( (ec + cos(vk) ) / ( 1 + ec * cos(vk) ) );
double uk = omega + vk; //Argument of latitude (rad)
double rk = A * ( 1 - ec * cos(Ek) ); //Corrected radius (meter)
double ik = io + del_ik; //Corrected inclination (rad)
double xk1 = rk * cos(uk); //x position in orital plane (meter)
double yk1 = rk * sin(uk); //y position in orital plane (meter)
double OMEGAk = OMEGAo + ( (OMEGAdot - OMEGAdote) * tk ) - (OMEGAdote * toa); //Corrected longitude of ascending node (rad)

//Calculations for ECEF coordinates
double xk = ( xk1 * cos(OMEGAk) ) - ( yk1 * cos(ik) * sin(OMEGAk) ); //Satellite x ECEF coordinate (meter)
double yk = ( xk1 * sin(OMEGAk) ) + ( yk1 * cos(ik) * cos(OMEGAk) ); //Satellite y ECEF coordinate (meter)
double zk = yk1 * sin(ik); //Satellite z ECEF coordinate (meter)

//Convert ECEF coordinates to East-North-Up (ENU) coordinates
double East = -sin(long_rad) * (xk - xLocalRef) + cos(long_rad) * (yk - yLocalRef);
double North = ( -sin(lat_rad) * cos(long_rad) * (xk - xLocalRef) ) - ( sin(lat_rad) * sin(long_rad) * (yk - yLocalRef) ) + ( cos(lat_rad) * (zk - zLocalRef) );
double Up = ( cos(lat_rad) * cos(long_rad) * (xk - xLocalRef) ) + ( cos(lat_rad) * sin(long_rad) * (yk - yLocalRef) ) + ( sin(lat_rad) * (zk - zLocalRef) );
EAST[i1] = East;
NORTH[i1] = North;
UP[i1] = Up;
//End of Position Calculations of Satellites

//Determine Line of Sight between Target and Satellite
```

```
		double mag_Tgt = sqrt( pow(xTgt, 2) + pow(yTgt, 2) + pow(zTgt, 2) ); //Distance of
Target from Earth center (meter)
		double xTgttoSV = xk - xTgt;
		double yTgttoSV = yk - yTgt;
		double zTgttoSV = zk - zTgt;
		double mag_TgttoSV = sqrt( pow( (xk - xTgt), 2 ) + pow( (yk - yTgt), 2 ) + pow( (zk -
zTgt), 2 ) ); //Distance from Target to Satellite
		double AngleFromTgt = acos(((xTgttoSV*xTgt) + (yTgttoSV*yTgt) +
(zTgttoSV*zTgt))/(mag_TgttoSV*mag_Tgt));
		int Los;

		if (AngleFromTgt < (pi/2-(obs*pi/180)) && SVHEALTH == 0)
			Los = 1; //There is Line of Sight
		else
			Los = 0; //No Line of Sight

		LOS[i1] = Los;
		//End of Determination of Line of Sight
	}


	//Assign Visible Satellites
	int i2;
	int i3 = 0;
	double SV[31][3];

	for(i2=0; i2<NumSV; i2++)
	{
		if (LOS[i2] == 1)
		{
			//Assigning coordinates to respective visible satellites
			SV[i3][0] = EAST[i2];
			SV[i3][1] = NORTH[i2];
			SV[i3][2] = UP[i2];
			i3++;
		}
	}
```

```
int NumVisibleSV = i3;

        //---------------------------------------------------------------------
        //Calculate DOP for Specified Time
        //---------------------------------------------------------------------
        //Pseudo-Range and Directional Derivative Loop
        int i4;
        double r[31], Dx[31], Dy[31], Dz[31], Dt[31];

        for(i4=0; i4<NumVisibleSV; i4++)
        {
                //Calculate pseudo-ranges from target position to visible satellites
                r[i4] = sqrt( pow(SV[i4][0], 2) + pow(SV[i4][1], 2) + pow((SV[i4][2] - alt), 2) );
                //Calculate directional derivatives for East, North, Up and Time
                Dx[i4] = SV[i4][0] / r[i4];
                Dy[i4] = SV[i4][1] / r[i4];
                Dz[i4] = (SV[i4][2] - alt) / r[i4];
                Dt[i4] = -1;
        }

        //Produce the Covariance Matrix from the Directional Derivatives
        int i5, i6;
        double Alp[31][4];

        for (i5=0; i5<31; i5++)
        {
                for (i6=0; i6<4; i6++)
                {
                        Alp[i5][i6] = 0; //Initialize Alp
                }
        }

        for (i5=0; i5<NumVisibleSV; i5++)
        {
                Alp[i5][0] = Dx[i5];
```

```
		Alp[i5][1] = Dy[i5];
		Alp[i5][2] = Dz[i5];
		Alp[i5][3] = Dt[i5];
}

//Transpose Alp to get Brv
int i7, i8;
double Brv[4][31];

for (i7=0; i7<4; i7++)
{
		for (i8=0; i8<31; i8++)
		{
				Alp[i5][i6] = 0; //Initialize Brv
		}
}

for (i7=0; i7<4; i7++)
{
		i8 = 0;
		while (i8<NumVisibleSV)
		{
				Brv[i7][i8] = Alp[i8][i7];
				i8++;
		}
}

//Matrix multiplication of Brv and Alp
int i9, i10, i11;
double Chl[4][4];

for (i9=0; i9<4; i9++)
{
		for (i10=0; i10<4; i10++)
		{
				Chl[i9][i10] = 0; //Initialize Chl
```

```
		}
}

for (i9=0; i9<4; i9++)
{
		for (i10=0; i10<4; i10++)
		{
			for (i11=0; i11<31; i11++)
			Chl[i9][i10] = Chl[i9][i10] + Brv[i9][i11]*Alp[i11][i10];
		}
}

//Inverse Chl
Matrix m(4,4);

int i12, i13;

for (i12=0; i12<4; i12++)
{
		for (i13=0; i13<4; i13++)
		{
			m(i12+1, i13+1) = Chl[i12][i13]; //Assign Chl to matrix, m
		}
}

Matrix minv(4,4);
double det=Matrix::inv(m,minv);
double Dlt[4][4];

for (i12=0; i12<4; i12++)
{
		for (i13=0; i13<4; i13++)
		{
			Dlt[i12][i13] = minv(i12+1, i13+1); //Assign inversed matrix, minv, to Dlt
		}
}
```

```
//Calculate DOP
xdop = sqrt(Dlt[0][0]);
ydop = sqrt(Dlt[1][1]);
vdop = sqrt(Dlt[2][2]);
tdop = sqrt(Dlt[3][3]);
hdop = sqrt(Dlt[0][0] + Dlt[1][1]);
pdop = sqrt(Dlt[0][0] + Dlt[1][1] + Dlt[2][2]);
gdop = sqrt(Dlt[0][0] + Dlt[1][1] + Dlt[2][2] + Dlt[3][3]);

//Display DOP to dialogue box
xdop0.Format((CString)"%.3f", xdop);
ydop0.Format((CString)"%.3f", ydop);
vdop0.Format((CString)"%.3f", vdop);
tdop0.Format((CString)"%.3f", tdop);
hdop0.Format((CString)"%.3f", hdop);
pdop0.Format((CString)"%.3f", pdop);
gdop0.Format((CString)"%.3f", gdop);

UpdateData(FALSE);
//End of DOP Calculation for Specified Time

//-------------------------------------------------------------------------
//Calculate DOP for 24hr Period (hourly sampling)
//-------------------------------------------------------------------------
double xdop2, ydop2, zdop2, tdop2, gdop2, pdop2, hdop2;// DOP variables for graph plotting

//Check if there is any existing graph. If so, delete it.
if(m_pPlotItems !=NULL)
{
        delete []m_pPlotItems;
        m_pPlotItems=NULL;
}
if(m_lpfs !=NULL)
{
        m_pGraph->ClearFunction();
```

```
            delete m_lpfs;
        }


        //Loop to calculate DOP values every 1hr
        double hr1 = 0;
        double mi1 = 0;

        num_items = 24; //number of plot points on the x-axis

        XDOP_PlotPoints=new double[num_items*2];
        YDOP_PlotPoints=new double[num_items*2];
        VDOP_PlotPoints=new double[num_items*2];
        TDOP_PlotPoints=new double[num_items*2];
        HDOP_PlotPoints=new double[num_items*2];
        PDOP_PlotPoints=new double[num_items*2];
        GDOP_PlotPoints=new double[num_items*2];

        //Loop for hourly sampling
        for(int i14=0; i14<=num_items*2; i14+=2)
        {
            JD = 367*yr - floor(7*(yr + floor((mth + 9)/12))/4) - floor(3*(floor((yr + (mth-9)/7)/100)
+ 1)/4) + floor(275*mth/9) + day + 1721028.5;
            double fJD = (mi1/1440) + (hr1/24);
            JD = JD + fJD - (timezone + daylightsav)/24;

            //Correct for leap seconds up till Mar 2009
            int GpsLeapSec;

            if    ((JD>=2444786.5)&&(JD<2445151.5))
                GpsLeapSec=1;
            else if ((JD>=2445151.5)&&(JD<2445516.5))
                GpsLeapSec=2;
            else if ((JD>=2445516.5)&&(JD<2446247.5))
                GpsLeapSec=3;
            else if ((JD>=2446247.5)&&(JD<2447161.5))
```

```
        GpsLeapSec=4;
else if ((JD>=2447161.5)&&(JD<2447892.5))
        GpsLeapSec=5;
else if ((JD>=2447892.5)&&(JD<2448257.5))
        GpsLeapSec=6;
else if ((JD>=2448257.5)&&(JD<2448804.5))
        GpsLeapSec=7;
else if ((JD>=2448804.5)&&(JD<2449169.5))
        GpsLeapSec=8;
else if ((JD>=2449169.5)&&(JD<2449534.5))
        GpsLeapSec=9;
else if ((JD>=2449534.5)&&(JD<2450083.5))
        GpsLeapSec=10;
else if ((JD>=2450083.5)&&(JD<2450630.5))
        GpsLeapSec=11;
else if ((JD>=2450630.5)&&(JD<2451179.5))
        GpsLeapSec=12;
else if ((JD>=2451179.5)&&(JD<2453371.5))
        GpsLeapSec=13;
else if ((JD>=2453371.5)&&(JD<2454466.5))
        GpsLeapSec=14;
else GpsLeapSec=15; //Add when new seconds are added

JD = JD + GpsLeapSec/sec_per_day;
double gps_week = floor( (JD - JAN61980)/7 );
double sec_of_week = ( (JD - JAN61980) - gps_week * 7 ) * sec_per_day;
//End of GPS time conversion

while (gps_week >= 1024)
{
        gps_week = gps_week - 1024;
}

//Reads the almanac data from specified location
ifstream in(almanac_location, ios::in); //opens the almanac file
in >> NumSV >> inputline >> wn >> toa; //reads time info from almanac file
```

```
double tk = (gps_week - wn) * 604800 + (sec_of_week - toa); //Time since toa (sec)
(range: -302400 to 302400)

//Need to increase the array size if there are more than 31 GPS satellites
int PRN;
int SVN;
int URA;
double ec;
double del_ik;
double OMEGAdot;
double sqrtA;
double OMEGAo;
double omega;
double Mo;
double Af0;
double Af1;
double SVHEALTH;
double SVCONFIG;
double EAST[31], NORTH[31], UP[31];
int LOS[31];

int i1;
for (i1=0; i1<NumSV; i1++) //reads data for different satellites
{
        //Read the data from Almanac file
        in >> PRN >> SVN >> URA;
        in >> ec >> del_ik >> OMEGAdot;
        in >> sqrtA >> OMEGAo >> omega;
        in >> Mo >> Af0 >> Af1;
        in >> SVHEALTH >> SVCONFIG;

        //Conversion of some almanac data to radians
        del_ik = del_ik * pi;
        OMEGAdot = OMEGAdot * pi;
        OMEGAo = OMEGAo * pi;
```

```
omega = omega * pi;
Mo = Mo * pi;

//Position of Satellite Calculations
double A = pow(sqrtA, 2);
double n = sqrt(mju / pow(A, 3));
double Mk = Mo + (tk * n); //Mean anomaly (rad)

//Start values for iterative solution of Kepler Equation
double Ek = Mk;
double Eold = 0;

while ( abs(Ek - Eold) >= 1e-10 )
{
	Eold = Ek;
	Ek = Mk + ec*sin(Ek);
}

double vk = atan2( ( sqrt(1 - pow(ec, 2) ) * sin(Ek) )/( 1 - ec * cos(Ek) ), ( cos(Ek) - ec )/( 1 - ec * cos(Ek) ) ); //True Anomaly (rad)
Ek = acos( (ec + cos(vk) ) / ( 1 + ec * cos(vk) ) );
double uk = omega + vk; //Argument of latitude (rad)
double rk = A * ( 1 - ec * cos(Ek) ); //Corrected radius (meter)
double ik = io + del_ik; //Corrected inclination (rad)
double xk1 = rk * cos(uk); //x position in orital plane (meter)
double yk1 = rk * sin(uk); //y position in orital plane (meter)
double OMEGAk = OMEGAo + ( (OMEGAdot - OMEGAdote) * tk ) - (OMEGAdote * toa); //Corrected longitude of ascending node (rad)

//Calculations for ECEF coordinates
double xk = ( xk1 * cos(OMEGAk) ) - ( yk1 * cos(ik) * sin(OMEGAk) ); //Satellite x ECEF coordinate (meter)
double yk = ( xk1 * sin(OMEGAk) ) + ( yk1 * cos(ik) * cos(OMEGAk) ); //Satellite y ECEF coordinate (meter)
double zk = yk1 * sin(ik); //Satellite z ECEF coordinate (meter)

//Convert ECEF coordinates to East-North-Up (ENU) coordinates
```

```
                double East = -sin(long_rad) * (xk - xLocalRef) + cos(long_rad) * (yk -
yLocalRef);
                double North = ( -sin(lat_rad) * cos(long_rad) * (xk - xLocalRef) ) - (
sin(lat_rad) * sin(long_rad) * (yk - yLocalRef) ) + ( cos(lat_rad) * (zk - zLocalRef) );
                double Up = ( cos(lat_rad) * cos(long_rad) * (xk - xLocalRef) ) + ( cos(lat_rad)
* sin(long_rad) * (yk - yLocalRef) ) + ( sin(lat_rad) * (zk - zLocalRef) );


                EAST[i1] = East;
                NORTH[i1] = North;
                UP[i1] = Up;
                //End of Position of Satellite Calculations



                //Determine Line of Sight between Target and Satellite
                double mag_Tgt = sqrt( pow(xTgt, 2) + pow(yTgt, 2) + pow(zTgt, 2) );
//Distance of Target from Earth center (meter)
                double mag_SV = sqrt( pow(xk, 2) + pow(yk, 2) + pow(zk, 2) ); //Distance of
Satellite from Earth center (meter)
                double AngleTOS = acos( ( (xk * xTgt) + (yk * yTgt) + (zk * zTgt) ) / (mag_Tgt
* mag_SV) ); //Angle between Target and Satellite with origin at center of Earth (rad)
                double mag_SVproj = mag_SV * cos(AngleTOS); //Magnitude of projection of
the Satellite vector onto Target vector (meter)
                double xTgttoSV = xk - xTgt;
                double yTgttoSV = yk - yTgt;
                double zTgttoSV = zk - zTgt;
                double xTgttoSVproj = ((mag_SVproj - mag_Tgt) / mag_Tgt) * xTgt;
                double yTgttoSVproj = ((mag_SVproj - mag_Tgt) / mag_Tgt) * yTgt;
                double zTgttoSVproj = ((mag_SVproj - mag_Tgt) / mag_Tgt) * zTgt;
                double mag_TgttoSV = sqrt( pow( (xk - xTgt), 2 ) + pow( (yk - yTgt), 2 ) +
pow( (zk - zTgt), 2 ) ); //Distance from Target to Satellite
                double AngleFromTgt = acos( ( (xTgttoSV * xTgttoSVproj) + (yTgttoSV *
yTgttoSVproj) + (zTgttoSV * zTgttoSVproj) ) / ( (mag_SVproj - mag_Tgt) * mag_TgttoSV ) ); //Angle
between Target and Satellite with origin at Target (rad)
                int Los;


                if ( mag_SVproj > mag_Tgt && AngleTOS < (pi/2) && AngleFromTgt < ( pi/2
- (obs * pi/180) ) )
                        Los = 1; //There is Line of Sight
                else
                        Los = 0; //No Line of Sight
```

```
        LOS[i1] = Los;
        //End of Determination of Line of Sight
}


//Assign Visible Satellites
int i2;
int i3 = 0;
double SV[31][3];

for(i2=0; i2<NumSV; i2++)
{
        if (LOS[i2] == 1)
        {
                //Assigning coordinates to respective visible satellites
                SV[i3][0] = EAST[i2];
                SV[i3][1] = NORTH[i2];
                SV[i3][2] = UP[i2];
                i3++;
        }
}

int NumVisibleSV = i3;

//---------------------------------------------------------------------
//Calculate DOP
//---------------------------------------------------------------------
//Pseudo-Range and Directional Derivative Loop
int i4;
double r[31], Dx[31], Dy[31], Dz[31], Dt[31];

for(i4=0; i4<NumVisibleSV; i4++)
{
        //Calculate pseudo-ranges from target position to visible satellites
        r[i4] = sqrt( pow(SV[i4][0], 2) + pow(SV[i4][1], 2) + pow((SV[i4][2] - alt), 2) );
```

```
        //Calculate directional derivatives for East, North, Up and Time
        Dx[i4] = SV[i4][0] / r[i4];
        Dy[i4] = SV[i4][1] / r[i4];
        Dz[i4] = (SV[i4][2] - alt) / r[i4];
        Dt[i4] = -1;
}

//Produce the Covariance Matrix from the Directional Derivatives
int i5, i6;
double Alp[31][4];

for (i5=0; i5<31; i5++)
{
        for (i6=0; i6<4; i6++)
        {
                Alp[i5][i6] = 0; //Initialize Alp
        }
}

for (i5=0; i5<NumVisibleSV; i5++)
{
        Alp[i5][0] = Dx[i5];
        Alp[i5][1] = Dy[i5];
        Alp[i5][2] = Dz[i5];
        Alp[i5][3] = Dt[i5];
}

//Transpose Alp to get Brv
int i7, i8;
double Brv[4][31];

for (i7=0; i7<4; i7++)
{
        for (i8=0; i8<31; i8++)
        {
                Alp[i5][i6] = 0; //Initialize Brv
```

```
        }
}

for (i7=0; i7<4; i7++)
{
        i8 = 0;
        while (i8<NumVisibleSV)
        {
                Brv[i7][i8] = Alp[i8][i7];
                i8++;
        }
}

//Matrix multiplication of Brv and Alp
int i9, i10, i11;
double Chl[4][4];

for (i9=0; i9<4; i9++)
{
        for (i10=0; i10<4; i10++)
        {
                Chl[i9][i10] = 0; //Initialize Chl
        }
}

for (i9=0; i9<4; i9++)
{
        for (i10=0; i10<4; i10++)
        {
                for (i11=0; i11<31; i11++)
                Chl[i9][i10] = Chl[i9][i10] + Brv[i9][i11]*Alp[i11][i10];
        }
}

//Inverse Chl
Matrix m(4,4);
```

```
int i12, i13;

for (i12=0; i12<4; i12++)
{
        for (i13=0; i13<4; i13++)
        {
                m(i12+1, i13+1) = Chl[i12][i13]; //Assign Chl to matrix, m
        }
}

Matrix minv(4,4);
double det=Matrix::inv(m,minv);
double Dlt[4][4];

for (i12=0; i12<4; i12++)
{
        for (i13=0; i13<4; i13++)
        {
                Dlt[i12][i13] = minv(i12+1, i13+1); //Assign inversed matrix, minv, to
Dlt
        }
}

//Calculate DOP
xdop2 = sqrt(Dlt[0][0]);
ydop2 = sqrt(Dlt[1][1]);
zdop2 = sqrt(Dlt[2][2]);
tdop2 = sqrt(Dlt[3][3]);
hdop2 = sqrt(Dlt[0][0] + Dlt[1][1]);
pdop2 = sqrt(Dlt[0][0] + Dlt[1][1] + Dlt[2][2]);
gdop2 = sqrt(Dlt[0][0] + Dlt[1][1] + Dlt[2][2] + Dlt[3][3]);

//Assign x and y values for graph plotting
XDOP_PlotPoints[i14]=hr1; //assign x-values
XDOP_PlotPoints[i14+1]= xdop2; //assign y-values
```

```
            YDOP_PlotPoints[i14]=hr1;
            YDOP_PlotPoints[i14+1]= ydop2;

            VDOP_PlotPoints[i14]=hr1;
            VDOP_PlotPoints[i14+1]= zdop2;

            TDOP_PlotPoints[i14]=hr1;
            TDOP_PlotPoints[i14+1]= tdop2;

            HDOP_PlotPoints[i14]=hr1;
            HDOP_PlotPoints[i14+1]= hdop2;

            PDOP_PlotPoints[i14]=hr1;
            PDOP_PlotPoints[i14+1]= pdop2;

            GDOP_PlotPoints[i14]=hr1;
            GDOP_PlotPoints[i14+1]= gdop2;

            hr1++;
}

got_data = true;
//End of DOP calculations for 24hr period

//-------------------------------------------------------------------------
//Plot Graph for HDOP as default
//-------------------------------------------------------------------------
m_pPlotItems = new double[num_items*2];

//Initialize the plot points
m_lpfs = NULL;
m_pPlotItems = NULL;

m_pPlotItems = HDOP_PlotPoints; //Assign plot points
```

```
	//Set the graph position (left-right, up-down) and size (width, height)
	m_pGraph->SetGraphSizePos(230,230,0,0);

	m_lpfs =new G_FUNCTIONSTRUCT;
	memset(m_lpfs,0,sizeof(G_FUNCTIONSTRUCT));
	m_lpfs->FuncType=G_PLOTXY;
	m_lpfs->szGraphTitle="Variation in HDOP over 24hr Period";
	m_lpfs->xMax=24;
	m_lpfs->xMin=0;
	m_lpfs->yMax=5;
	m_lpfs->yMin=0;
	m_lpfs->ChartType=G_LINECHART;
	m_lpfs->pPlotXYItems=m_pPlotItems;
	m_lpfs->num_PlotXYItems=num_items;
	m_pGraph->DoFunction(m_lpfs);
	//End of graph plotting

	//Set display to indicate HDOP is checked
	((CButton*)GetDlgItem(IDC_XDOP_RADIO))->SetCheck(0);
	((CButton*)GetDlgItem(IDC_YDOP_RADIO))->SetCheck(0);
	((CButton*)GetDlgItem(IDC_VDOP_RADIO))->SetCheck(0);
	((CButton*)GetDlgItem(IDC_TDOP_RADIO))->SetCheck(0);
	((CButton*)GetDlgItem(IDC_PDOP_RADIO))->SetCheck(0);
	((CButton*)GetDlgItem(IDC_GDOP_RADIO))->SetCheck(0);
	((CButton*)GetDlgItem(IDC_HDOP_RADIO))->SetCheck(1);
}


void CDOP_CalculatorDlg::OnBnClickedXdopRadio()
{
	if (got_data == true)
	{
		//Initialize the plot points
		m_lpfs = NULL;
		m_pPlotItems = NULL;
```

```
		m_pPlotItems = XDOP_PlotPoints; //Assign plot points

		//Set the graph position (left-right, up-down) and size (width, height)
		m_pGraph->SetGraphSizePos(230,230,0,0);

		m_lpfs =new G_FUNCTIONSTRUCT;
		memset(m_lpfs,0,sizeof(G_FUNCTIONSTRUCT));
		m_lpfs->FuncType=G_PLOTXY;
		m_lpfs->szGraphTitle="Variation in XDOP over 24hr Period";
		m_lpfs->xMax=24;
		m_lpfs->xMin=0;
		m_lpfs->yMax=5;
		m_lpfs->yMin=0;
		m_lpfs->ChartType=G_LINECHART;
		m_lpfs->pPlotXYItems=m_pPlotItems;
		m_lpfs->num_PlotXYItems=num_items;
		m_pGraph->DoFunction(m_lpfs);
		//End of graph plotting
	}
	else
	{
		AfxMessageBox((CString)"Error: No data to plot graph. Press 'Compute'
Button.",MB_ICONERROR);
	}
}


void CDOP_CalculatorDlg::OnBnClickedYdopRadio()
{
	if (got_data == true)
	{
		//Initialize the plot points
		m_lpfs = NULL;
		m_pPlotItems = NULL;

		m_pPlotItems = YDOP_PlotPoints; //Assign plot points
```

```
		//Set the graph position (left-right, up-down) and size (width, height)
		m_pGraph->SetGraphSizePos(230,230,0,0);

		m_lpfs =new G_FUNCTIONSTRUCT;
		memset(m_lpfs,0,sizeof(G_FUNCTIONSTRUCT));
		m_lpfs->FuncType=G_PLOTXY;
		m_lpfs->szGraphTitle="Variation in YDOP over 24hr Period";
		m_lpfs->xMax=24;
		m_lpfs->xMin=0;
		m_lpfs->yMax=5;
		m_lpfs->yMin=0;
		m_lpfs->ChartType=G_LINECHART;
		m_lpfs->pPlotXYItems=m_pPlotItems;
		m_lpfs->num_PlotXYItems=num_items;
		m_pGraph->DoFunction(m_lpfs);
		//End of graph plotting
	}
	else
	{
		AfxMessageBox((CString)"Error: No data to plot graph. Press 'Compute'
Button.",MB_ICONERROR);
	}
}


void CDOP_CalculatorDlg::OnBnClickedVdopRadio()
{
	if (got_data == true)
	{
		//Initialize the plot points
		m_lpfs = NULL;
		m_pPlotItems = NULL;

		m_pPlotItems = VDOP_PlotPoints; //Assign plot points
```

```
		//Set the graph position (left-right, up-down) and size (width, height)
		m_pGraph->SetGraphSizePos(230,230,0,0);

		m_lpfs =new G_FUNCTIONSTRUCT;
		memset(m_lpfs,0,sizeof(G_FUNCTIONSTRUCT));
		m_lpfs->FuncType=G_PLOTXY;
		m_lpfs->szGraphTitle="Variation in VDOP over 24hr Period";
		m_lpfs->xMax=24;
		m_lpfs->xMin=0;
		m_lpfs->yMax=5;
		m_lpfs->yMin=0;
		m_lpfs->ChartType=G_LINECHART;
		m_lpfs->pPlotXYItems=m_pPlotItems;
		m_lpfs->num_PlotXYItems=num_items;
		m_pGraph->DoFunction(m_lpfs);
		//End of graph plotting
	}
	else
	{
		AfxMessageBox((CString)"Error: No data to plot graph. Press 'Compute'
Button.",MB_ICONERROR);
	}
}


void CDOP_CalculatorDlg::OnBnClickedTdopRadio()
{
	if (got_data == true)
	{
		//Initialize the plot points
		m_lpfs = NULL;
		m_pPlotItems = NULL;

		m_pPlotItems = TDOP_PlotPoints; //Assign plot points

		//Set the graph position (left-right, up-down) and size (width, height)
```

```
		m_pGraph->SetGraphSizePos(230,230,0,0);

		m_lpfs =new G_FUNCTIONSTRUCT;
		memset(m_lpfs,0,sizeof(G_FUNCTIONSTRUCT));
		m_lpfs->FuncType=G_PLOTXY;
		m_lpfs->szGraphTitle="Variation in TDOP over 24hr Period";
		m_lpfs->xMax=24;
		m_lpfs->xMin=0;
		m_lpfs->yMax=5;
		m_lpfs->yMin=0;
		m_lpfs->ChartType=G_LINECHART;
		m_lpfs->pPlotXYItems=m_pPlotItems;
		m_lpfs->num_PlotXYItems=num_items;
		m_pGraph->DoFunction(m_lpfs);
		//End of graph plotting
	}
	else
	{
		AfxMessageBox((CString)"Error: No data to plot graph. Press 'Compute'
Button.",MB_ICONERROR);
	}
}


void CDOP_CalculatorDlg::OnBnClickedHdopRadio()
{
	if (got_data == true)
	{
		//Initialize the plot points
		m_lpfs = NULL;
		m_pPlotItems = NULL;

		m_pPlotItems = HDOP_PlotPoints; //Assign plot points

		//Set the graph position (left-right, up-down) and size (width, height)
		m_pGraph->SetGraphSizePos(230,230,0,0);
```

```
		m_lpfs =new G_FUNCTIONSTRUCT;
		memset(m_lpfs,0,sizeof(G_FUNCTIONSTRUCT));
		m_lpfs->FuncType=G_PLOTXY;
		m_lpfs->szGraphTitle="Variation in HDOP over 24hr Period";
		m_lpfs->xMax=24;
		m_lpfs->xMin=0;
		m_lpfs->yMax=5;
		m_lpfs->yMin=0;
		m_lpfs->ChartType=G_LINECHART;
		m_lpfs->pPlotXYItems=m_pPlotItems;
		m_lpfs->num_PlotXYItems=num_items;
		m_pGraph->DoFunction(m_lpfs);
		//End of graph plotting
	}
	else
	{
		AfxMessageBox((CString)"Error: No data to plot graph. Press 'Compute'
Button.",MB_ICONERROR);
	}
}


void CDOP_CalculatorDlg::OnBnClickedPdopRadio()
{
	if (got_data == true)
	{
		//Initialize the plot points
		m_lpfs = NULL;
		m_pPlotItems = NULL;

		m_pPlotItems = PDOP_PlotPoints; //Assign plot points

		//Set the graph position (left-right, up-down) and size (width, height)
		m_pGraph->SetGraphSizePos(230,230,0,0);
```

```
		m_lpfs =new G_FUNCTIONSTRUCT;
		memset(m_lpfs,0,sizeof(G_FUNCTIONSTRUCT));
		m_lpfs->FuncType=G_PLOTXY;
		m_lpfs->szGraphTitle="Variation in PDOP over 24hr Period";
		m_lpfs->xMax=24;
		m_lpfs->xMin=0;
		m_lpfs->yMax=5;
		m_lpfs->yMin=0;
		m_lpfs->ChartType=G_LINECHART;
		m_lpfs->pPlotXYItems=m_pPlotItems;
		m_lpfs->num_PlotXYItems=num_items;
		m_pGraph->DoFunction(m_lpfs);
		//End of graph plotting
	}
	else
	{
		AfxMessageBox((CString)"Error: No data to plot graph. Press 'Compute'
Button.",MB_ICONERROR);
	}
}


void CDOP_CalculatorDlg::OnBnClickedGdopRadio()
{
	if (got_data == true)
	{
		//Initialize the plot points
		m_lpfs = NULL;
		m_pPlotItems = NULL;

		m_pPlotItems = GDOP_PlotPoints; //Assign plot points

		//Set the graph position (left-right, up-down) and size (width, height)
		m_pGraph->SetGraphSizePos(230,230,0,0);

		m_lpfs =new G_FUNCTIONSTRUCT;
```

```
                memset(m_lpfs,0,sizeof(G_FUNCTIONSTRUCT));
                m_lpfs->FuncType=G_PLOTXY;
                m_lpfs->szGraphTitle="Variation in GDOP over 24hr Period";
                m_lpfs->xMax=24;
                m_lpfs->xMin=0;
                m_lpfs->yMax=5;
                m_lpfs->yMin=0;
                m_lpfs->ChartType=G_LINECHART;
                m_lpfs->pPlotXYItems=m_pPlotItems;
                m_lpfs->num_PlotXYItems=num_items;
                m_pGraph->DoFunction(m_lpfs);
                //End of graph plotting
        }
        else
        {
                AfxMessageBox((CString)"Error: No data to plot graph. Press 'Compute'
Button.",MB_ICONERROR);
        }
}
```

## B. DOP_CALCULATORDLG.H

```
// DOP_CalculatorDlg.h : header file

#include "Graph.h"
#pragma once

// CDOP_CalculatorDlg dialog
class CDOP_CalculatorDlg : public CDialog
{
// Construction
public:
	double *m_pPlotItems;
	LPG_FUNCTIONSTRUCT m_lpfs;
	CGraph *m_pGraph;

	CDOP_CalculatorDlg(CWnd* pParent = NULL);	// standard constructor

// Dialog Data
	enum { IDD = IDD_DOP_CALCULATOR_DIALOG };

	protected:
	virtual void DoDataExchange(CDataExchange* pDX);	// DDX/DDV support

// Implementation
protected:
	HICON m_hIcon;

	// Generated message map functions
	virtual BOOL OnInitDialog();
	afx_msg void OnSysCommand(UINT nID, LPARAM lParam);
	afx_msg void OnPaint();
	afx_msg HCURSOR OnQueryDragIcon();
	DECLARE_MESSAGE_MAP()
public:
	double yr;
```

```
double mth;
double day;
double hr;
double mi;
double timezone;
double longi;
double lat;
double alt;
double obs;
BOOL daylightsaving;

CString xdop0;
CString ydop0;
CString vdop0;
CString tdop0;
CString hdop0;
CString pdop0;
CString gdop0;

afx_msg void OnBnClickedAlmanac();
afx_msg void OnBnClickedCompute();
afx_msg void OnBnClickedXdopRadio();
afx_msg void OnBnClickedYdopRadio();
afx_msg void OnBnClickedVdopRadio();
afx_msg void OnBnClickedTdopRadio();
afx_msg void OnBnClickedHdopRadio();
afx_msg void OnBnClickedPdopRadio();
afx_msg void OnBnClickedGdopRadio();

//Variables not automatically generated by Visual C++
CString almanac_location;
int num_items;
double daylightsav;
bool got_data;

double *XDOP_PlotPoints;
```

```
        double *YDOP_PlotPoints;
        double *VDOP_PlotPoints;
        double *TDOP_PlotPoints;
        double *HDOP_PlotPoints;
        double *PDOP_PlotPoints;
        double *GDOP_PlotPoints;

        double xdop;
        double ydop;
        double vdop;
        double tdop;
        double hdop;
        double pdop;
        double gdop;
};
```

## C. GRAPH.CPP [AFTER 20]

```
// Graph.cpp: implementation of the CGraph class.

#include "stdafx.h"
#include <math.h>
#include "DOP_Calculator.h"
#include "Graph.h"

#ifdef _DEBUG
#undef THIS_FILE
static char THIS_FILE[]=__FILE__;
#define new DEBUG_NEW
#endif

#pragma warning (disable:4244 4018 4701)

CGraph::CGraph()
{
	GraphSetAllDefaults();
}

////////////////////// Constructor //////////////////////////////////
/*

*/

//////////////////////////////////////////////////////////////////////
CGraph::CGraph(CWnd *pParentWnd, int xPos, int yPos, int Width, int Height, UINT colorscheme)
{
	//First set everything to their default values
	GraphSetAllDefaults();

	//setup default values
	m_pWnd=pParentWnd;
	SetColorScheme(colorscheme);
	//set graph position
```

```
	m_iGraphX=xPos<0?0:xPos;
	m_iGraphY=yPos<0?0:yPos;
	//set graphsize (0 or less means default)
	m_iGraphWidth = Width < G_MINGRAPHWIDTH?G_MINGRAPHWIDTH:Width;
	m_iGraphHeight=Height < G_MINGRAPHHEIGHT?G_MINGRAPHHEIGHT:Height;

}
////////////////////// Default Destructor ////////////////////////////
/*

*/

//////////////////////////////////////////////////////////////////
CGraph::~CGraph()
{

}



////////////////////////// GraphSetAllDefaults ////////////////////////////////
/*	This function sets all the member variables to their default values creates the default font, etc....
	Because there are so may member variables, attempting to make constructors to cover all
	possibilites will be a pain - so all constructors should call this function first, then overwrite the
	default values as required.
*/

//////////////////////////////////////////////////////////////////////////
void CGraph::GraphSetAllDefaults()
{
	//setup default values
	m_pWnd=NULL;
	//create default font
	CreateGraphFont((const wchar_t *)"Courier",8);
	SetDefaultColorScheme();
	//m_bAutofit=TRUE;
	m_bShowGrid=TRUE;
	m_bShowTicks=TRUE;
	//set default graph position
```

```
	m_iGraphX=0;
	m_iGraphY=0;
	//set default graph size
	SetGraphSizePos(0, 0,G_MINGRAPHWIDTH,G_MINGRAPHHEIGHT);
	//set the axis scaling
	SetXAxisScale(G_DEFAULTXMIN,G_DEFAULTXMAX);
	SetYAxisScale(G_DEFAULTYMIN,G_DEFAULTYMAX);
	//set the legend and Title Texts
	SetDefaultGraphTitle();
	SetDefaultXLegend();

	//other stuff
	m_pFunctionParams=NULL;
}

////////////////// SetColorScheme ////////////////////////////////////////////
/*

	SetColorScheme set the color scheme for the graph
	Default colour scheme is the same as as the WHITE colorscheme (which is really grey)

*/

//////////////////////////////////////////////////////////////////////////////
void CGraph::SetColorScheme(int Scheme, BOOL bRedraw)
{
	/*
	This sets up the colors for various graph elements
	*/
	switch (Scheme)
	{
	case G_DEFAULTSCHEME:
	case G_WHITESCHEME:
		m_crYTickColor=RGB(0,0,0);
		m_crXTickColor=RGB(0,0,0);
		m_crYLegendTextColor=RGB(0,0,0);
		m_crXLegendTextColor=RGB(0,0,0);
```

```
                m_crGraphTitleColor=RGB(0,0,0);
                m_crGraphPenColor=RGB(0,0,0);
                m_crGraphBkColor=RGB(192,192,192);
                m_crGridColor=RGB(220,220,220);
                break;
        default:
                break;

        }
        if(bRedraw)
        {
                PaintGraph();
        }
}

////////////////////////////// CreateGraphFont ////////////////////////////////////
/*

        CreateGraphFont - Creates a font using the specified facename and point size


*/

//////////////////////////////////////////////////////////////////////////////
 void CGraph::CreateGraphFont(CString FaceName, UINT size)
 {
        //at this point we may not have a handle to a window - so to fill
        //things like char-width & height where we need a dc we will get a
         //whole screen dc

        CFont *poldfont;
        HDC hDC;
        CDC *dc;
        TEXTMETRIC textmetrics;
        if (!m_pWnd)//if we don't have a holding window
        {
                hDC=GetDC(0);//get a whole screen dc
```

```
        dc=new CDC;
        dc->Attach(hDC);
}
else
{
        dc=m_pWnd->GetDC();
}

//detach old font if any
m_GraphFont.Detach();
//create the new one
m_GraphFont.CreatePointFont(size*10,FaceName,dc);

poldfont=dc->SelectObject(&m_GraphFont);

dc->GetTextMetrics(&textmetrics);
m_iCharHeight=textmetrics.tmHeight;
m_iCharWidth=textmetrics.tmAveCharWidth;

dc->SelectObject(poldfont);
if(!m_pWnd)
{
        dc->Detach();
        ReleaseDC(0,hDC);
        delete dc;
}
else
{
        m_pWnd->ReleaseDC(dc);
}

m_iFontSize=size;
m_szFontFace=FaceName;

//we need to rescale the graph
SetXAxisScale(m_dXAxisMin, m_dXAxisMax);
```

```
        SetYAxisScale(m_dYAxisMin, m_dYAxisMax);
}

//////////////////////////   SetGraphSizePos ////////////////////////////////////////////////
/*

        Sets the size of the entire graph size and position with the holding parent window.
        A value of -1 for any parameter means don't change that parameter.
        A value of 0 for the Width or Height(or any value between 0 the minimum graph width or height)
        means use the default graph width or height
*/
////////////////////////////////////////////////////////////////////////////////
void CGraph::SetGraphSizePos(int xPos, int yPos, int Width, int Height)
{

        CRect rect;
        rect.left=m_iGraphX;
        rect.top=m_iGraphY;
        rect.right=rect.left+m_iGraphWidth;
        rect.bottom=rect.top+m_iGraphHeight;

        /*
        If xPos or yPos <=0 then  position will not be changed
        */
        m_iGraphX=xPos < 0?m_iGraphX:xPos;
        m_iGraphY=yPos < 0?m_iGraphY:yPos;

        /*
        A negative number or zero means no change of current width or height
        if a +ive size  given thats smaller than the default min size for
        height or width then use the default min size for those parameters;
        */
        Width= (Width>0 && Width <G_MINGRAPHWIDTH)?G_MINGRAPHWIDTH:Width;
        m_iGraphWidth = Width <= 0? m_iGraphWidth :Width;

        Height = (Height>0 && Height <G_MINGRAPHHEIGHT)?G_MINGRAPHHEIGHT:Height;
        m_iGraphHeight = Height <= 0? m_iGraphHeight : Height;
```

```
	//remember to rescale the graph
	SetXAxisScale(m_dXAxisMin,m_dXAxisMax);
	SetYAxisScale(m_dYAxisMin,m_dYAxisMax);

	//clear old graph from the screen
	if(m_pWnd)
	{
		m_pWnd->InvalidateRect(&rect,TRUE);
		m_pWnd->SendMessage(WM_PAINT,0,0);
	}
}


////////////////////////// CalcTopMargin ////////////////////////////////////
/*
	Returns the spacing between the top edge of the graph and the top of the plotting area.
*/
/////////////////////////////////////////////////////////////////////////////
UINT CGraph::CalcTopMargin()
{
	return 4*m_iCharHeight;
}


////////////////////  CalcBottommargin  ////////////////////////////////////
/*
	Returns the space between the bottom of the graph and the bottom of the actual plotting area
*/
/////////////////////////////////////////////////////////////////////////////
UINT CGraph::CalcBottomMargin()
{
	return 3*m_iCharHeight;
}

////////////////////// CalcLeftmargin ////////////////////////////////////
```

```
/*
	Returns the margin between the left of the graph and the left of the plotting area
*/
////////////////////////////////////////////////////////////////////////
UINT CGraph::CalcLeftMargin()
{
	return 4*m_iCharWidth;
}

////////////////////////// CalcRightMargin ////////////////////////////////////
/*
	Returns the space between the right side of the plotting area and the right side of the graph
*/
////////////////////////////////////////////////////////////////////////
UINT CGraph::CalcRightMargin()
{
	return m_iCharWidth;
}

////////////////////// SetXAxisScale //////////////////////////////////////
/*
	This sets the min and max values of the x-axis (of the plotting area)
	It also sets what proportion of the x-range is denoted by one pixel a.ka. PixelsPerX
	It also caluclates where the x-origin (the x==0) point lies along the x-axis - if x does not pass
	through zero the x-origin gidline is shown at the top or bottom of the graph.
	(the x-origin point on screen is stored in the m_iOriginX parameter as an offset from the LHS of
	the (total) graph
	26/2/2005 - The autofit parameter is always TRUE as scrolling is not yet implemented.
*/
////////////////////////////////////////////////////////////////////////
void CGraph::SetXAxisScale(double min, double max)
{
	//swap min and max if they are the wrong way around
	double temp,scale;
	if (max < min)
	{
		temp =min;
```

```
		min=max;
		max=temp;
}
//if min and max are the same (especially if they are both zero
//it can be a problem - so give them a bit of room
if(min==max)
{
		max+= 0.1;
}


//set the member variables
m_dXAxisMax=max;
m_dXAxisMin=min;

UINT lmargin=CalcLeftMargin();
UINT rmargin=CalcRightMargin();

if (m_bAutofit) //Autofit is always true @ 26 Feb 2005
{
		temp=max-min;//the spread of the x-axiz
		scale=(m_iGraphWidth-lmargin-rmargin)/temp;//calc pixels/x
		m_dPixelsPerX=scale;
}
else
{
		m_dPixelsPerX=1;//1:1
}

//where would the x-origin be located?
if( (min < 0) && (max >0))
{
		m_iOriginX=abs(min) * m_dPixelsPerX + lmargin;
}
else if ( (min <0) && (max <=0))
{
```

```
		m_iOriginX=(m_iGraphWidth)-rmargin;
	}
	else if (min >=0 && (max>=0))
	{
		m_iOriginX=lmargin;
	}
}

////////////////////// SetYAxisScale ////////////////////////////////////
/*
	This sets the min and max values of the y-axis (of the plotting area)
	It also sets what proportion of the grapgh axis pixels represents 1Y
	(PixelsPerY =plotheightinpixels/yrange)
	It also caluclates where the y-origin (the y==0) point lies along the y-axis - if y does not pass through zero the y-origin gidline is shown at the left or right of the graph (this can be overridden by using the SetYLineAtLeft() function.
	(the y-origin point on screen is stored in the m_iOriginY parameter as an offset from the bottom of the (total) graph
	26/2/2005 - The autofit parameter is always TRUE as scrolling is not yet implemented.
*/
////////////////////////////////////////////////////////////////////////
void CGraph::SetYAxisScale(double min, double max)
{
	double temp,scale;
	//swap min and max if they are the wrong way around
	if (max < min)
	{
		temp=min;
		min=max;
		max=temp;
	}

	//if min and max are the same (especially if they are both zero
	//it can be a problem - so give them a bit of room
	if(min==max)
	{
		max+= 0.1;
```

```
}

//set the member variables
m_dYAxisMin=min;
m_dYAxisMax=max;

//calculate scaling
UINT bmargin=CalcBottomMargin();
UINT tmargin=CalcTopMargin();
CRect dataarea=CalcDataArea();

if (m_bAutofit)//Always TRUE
{
        temp=max-min;//the spread of the y-axiz
        scale=(m_iGraphHeight-(bmargin+tmargin))/temp;//calc pixels/x
        m_dPixelsPerY=scale;
}
else
{
        m_dPixelsPerY=1;//1:1
}

//where should the Y origin be?

if (min<0 && max >0)//if Y passes through zero
{
        //from the bottom of the graph
        m_iOriginY=abs(min)*m_dPixelsPerY+bmargin;
}
else if(min<0 && max<=0)//if Y values are all negative
{
        m_iOriginY=(dataarea.bottom-dataarea.top)+bmargin;
}
else if (min >=0 && max >=0)//if Y values are all positive
{
        m_iOriginY=bmargin;
```

```
        }
}


//////////////////////// PaintGraph ////////////////////////////////////
/*
        This paints the entire graph on to the holding window's client area;
        It does it in steps starting from the background and working forward.
        As the graph is NOT a window object in it's own right, it uses the display context of the holding
        window. If it has not been given a pointer to the holding window, it will not paint.
        Any CGraph routine that paints to the screen, checks the window pointer first.
        The last thing to be painted is the plotting of the function data (if any)
*/
/////////////////////////////////////////////////////////////////////////
void CGraph::PaintGraph()
{
        //here we draw the graph
        //step 1: Draw the surrounding rectangle
        //for the whole graph
        if (m_pWnd==NULL)
        {
                return;
        }
        CRect rect;
        CPen pen, *oldpen;
        CDC *dc=m_pWnd->GetDC();
        //some useful calculations
        UINT lmargin=CalcLeftMargin();
        UINT rmargin=CalcRightMargin();
        UINT bmargin=CalcBottomMargin();
        UINT tmargin=CalcTopMargin();
        UINT Graphbottom=m_iGraphY+m_iGraphHeight;
        UINT Graphright=m_iGraphX+m_iGraphWidth;

        //step 2: color the background
        CBrush brush,*poldbrush;
```

```
brush.CreateSolidBrush(m_crGraphBkColor);
pen.CreatePen(PS_SOLID,1,m_crGraphBkColor);
rect.left=m_iGraphX;
rect.right =rect.left+m_iGraphWidth;
rect.top=m_iGraphY;
rect.bottom=rect.top+m_iGraphHeight;
oldpen=dc->SelectObject(&pen);
poldbrush=dc->SelectObject(&brush);
dc->Rectangle(&rect);
dc->SelectObject(oldpen);
dc->SelectObject(poldbrush);
pen.Detach();

//step 3: Draw Grid if required
DrawGrid();

//step 4: Draw Axes
//draw x-axis
dc->MoveTo(m_iGraphX+lmargin,Graphbottom-m_iOriginY);
pen.CreatePen(PS_SOLID,1,m_crXTickColor);
oldpen=dc->SelectObject(&pen);
dc->LineTo(Graphright-rmargin,Graphbottom-m_iOriginY);
dc->SelectObject(oldpen);
pen.Detach();

//draw the Y Axis
pen.CreatePen(PS_SOLID,1,m_crXTickColor);
oldpen=dc->SelectObject(&pen);
if (!m_bYLineAtLeft)
{
        //draw the Y Line so that it intercepts
        //the x-line like crosshairs
        dc->MoveTo(m_iGraphX+m_iOriginX,m_iGraphY+tmargin);
        dc->LineTo(m_iGraphX+m_iOriginX,Graphbottom-bmargin);
}
else
```

```
        {
                //draw the Y Line at the LHS
                dc->MoveTo(m_iGraphX+lmargin,m_iGraphY+tmargin);
                dc->LineTo(m_iGraphX+lmargin,Graphbottom-bmargin);
        }
        dc->SelectObject(oldpen);
        pen.Detach();

        //step 5: draw ticks
        DrawTicks();

        //step 6: Write Graph title
        DrawGraphTitle();

        //step 7: Write x-legend
        DrawXLegend();

        //step 8: write the x & y axes values
        DrawXAxisNumbers();
        DrawYAxisNumbers();

        //Step 9
        //draw Function
        DrawFunction();

        //Cleanup
        m_pWnd->ReleaseDC(dc);
}

////////////////////// DrawGrid ////////////////////////////////////////////////
/*
The grid comprises two parts - the rectangle drawn around the plotting areaand the vertical&horizontal gridlines.
The outline rectangle is always drawn, the drawing of the gidlines is contrlled by the m_bShowGrid member parameter using the ShowGrid function.
The grid color is  m_crGridColor.
*/
```

```
////////////////////////////////////////////////////////////////////////////
void CGraph::DrawGrid()
{

	if(!m_pWnd)
	{
		return;
	}

	/*
	Always draw the dataarea outline rectangle
	*/

	CRect dataarea=CalcDataArea();;//where the graph data is actually drawn

	//need a pen of colour m_crGridColor
	CPen pen, *poldpen;
	pen.CreatePen(PS_SOLID,1,m_crGridColor);

	CDC *pdc=m_pWnd->GetDC();
	//to make a rectangle outline we have to use a polyline
	//need an array of points
	//a recatangle comprises 5 point
	CPoint points[5];
	//topleft
	points[0].x=dataarea.left;
	points[0].y=dataarea.top;
	//topright
	points[1].x=dataarea.right;
	points[1].y=dataarea.top;
	//bottomright
	points[2].x=dataarea.right;
	points[2].y=dataarea.bottom;
	//leftbottom
	points[3].x=dataarea.left;
	points[3].y=dataarea.bottom;
```

```
//back to topleft
points[4].x=dataarea.left;
points[4].y=dataarea.top;

poldpen=pdc->SelectObject(&pen);
pdc->Polyline(points,5);//draw the outline rectangle

/* Now check whether the grid itself should
be shown
*/
if(!m_bShowGrid)
{
        pdc->SelectObject(poldpen);
        m_pWnd->ReleaseDC(pdc);
        return;
}

//draw the X-axis gridlines
//note x-axis grid lines run top tom bottom
double GridSpacing;
GridSpacing=CalcXAxisGridAndTicks();
int n; //for the loop
for(n=1; n<G_X_NUMTICKSANDGRID;n++)
{
        pdc->MoveTo(dataarea.left+GridSpacing*n , dataarea.top);
        pdc->LineTo(dataarea.left+GridSpacing*n,dataarea.bottom);
}
//do the Y grid lines
//note Y gridlines run left - right
GridSpacing=CalcYAxisGridAndTicks();
for(n=1; n<G_Y_NUMTICKSANDGRID;n++)
{
        pdc->MoveTo(dataarea.left , dataarea.top+GridSpacing*n);
        pdc->LineTo(dataarea.right,dataarea.top+GridSpacing*n);
}
```

```
	pdc->SelectObject(poldpen);
	m_pWnd->ReleaseDC(pdc);
}

////////////////////// DrawTicks ////////////////////////////////////////
/*
	DrawTicks does two things - it draws the x & Y axis lines and it also draws the little 'tick' lines.
	The axis lines are always shown but the 'ticks' are controlled by the m_bShowTicks member
	(using the ShowTicks function);
	The length of the ticks are set by the #define in the header file the axis lines and ticks use the same
	color (m_crXTickColor for the x-axis and m_crYTickColor for the y-axis)
*/
//*****************************************************************
void CGraph::DrawTicks()
{
	//pretty much the same as showing the grid because the ticks and
	//tick align
	if (!m_pWnd)
	{
		return;
	}
	if (!m_bShowTicks)
	{
		return;
	}
	CPen TickPen, *poldpen;

	CRect dataarea=CalcDataArea();
	UINT GraphBottom=m_iGraphY+m_iGraphHeight;

	CDC *pdc=m_pWnd->GetDC();
	//start with the x-axis
	TickPen.CreatePen(PS_SOLID,1,m_crXTickColor);
	poldpen=pdc->SelectObject(&TickPen);
	double GridSpacing=CalcXAxisGridAndTicks();
	//the ticklines vertically straddle the x-axis
	//two problems though - if the x-line is at or very close to the
```

```
	//top or bottom of the dataarea
	UINT xtoptick=((GraphBottom-m_iOriginY)-dataarea.top <
G_TICKLENGTH/2)?(GraphBottom-m_iOriginY)-dataarea.top:G_TICKLENGTH/2;
	UINT xbottick=(dataarea.bottom-(GraphBottom-m_iOriginY)
<G_TICKLENGTH/2)?dataarea.bottom-(GraphBottom-m_iOriginY):G_TICKLENGTH/2;
	int n;
	for(n=1;n<G_X_NUMTICKSANDGRID;n++)
	{
		//loop and do the ticks
		//topticks
		pdc->MoveTo(dataarea.left+n*GridSpacing,GraphBottom-m_iOriginY);
		pdc->LineTo(dataarea.left+n*GridSpacing,GraphBottom-m_iOriginY-xtoptick);
		//bottom ticks
		pdc->LineTo(dataarea.left+n*GridSpacing,GraphBottom-m_iOriginY+xbottick);

	}

	//now do the x axis ticks
	pdc->SelectObject(poldpen);
	TickPen.Detach();
	TickPen.CreatePen(PS_SOLID,1,m_crYTickColor);
	poldpen=pdc->SelectObject(&TickPen);
	GridSpacing=CalcYAxisGridAndTicks();
	//the tick horizontally straddle the the Y axis
	//some problems though - if the y-axis is at or very close to the
	//left or right of the data area or if the Y-line memeber is set to left
	//handside
	UINT ylefttick=( (m_iGraphX+m_iOriginX)-dataarea.left <G_TICKLENGTH/2)?
(m_iGraphX+m_iOriginX)-dataarea.left:G_TICKLENGTH/2;
	UINT yrighttick=( dataarea.right-(m_iGraphX+m_iOriginX) <G_TICKLENGTH/2)?
dataarea.right-(m_iGraphX+m_iOriginX):G_TICKLENGTH/2;
	//check for the special case where the y-axis has been forced to the left
	if(m_bYLineAtLeft)
	{
		ylefttick=0;
	}
```

```
	int x,y;
	if(m_bYLineAtLeft)
	{
		x=dataarea.left;
		y=dataarea.bottom;
	}
	else
	{
		x=m_iGraphX+m_iOriginX;
		y=dataarea.bottom;
	}


	for(n=1;n<G_Y_NUMTICKSANDGRID;n++)
	{
		//Loop and do the y axis ticks

		pdc->MoveTo(x,y-(n*GridSpacing));
		//do left side tick
		pdc->LineTo(x-ylefttick,y-(n*GridSpacing));
		//do rightside tick
		pdc->LineTo(x+yrighttick,y-(n*GridSpacing));
	}

	//cleanup
	pdc->SelectObject(poldpen);
	m_pWnd->ReleaseDC(pdc);
}


////////////// CalcDataArea ////////////////////////////////////////////////////
/*
	This function calculates the actual plotting area of the graph this is the graph area minus the
	top,bottom,left & right margins
	Returns: CRect with the plotting area (in client area co-ords)
*/
```

```
////////////////////////////////////////////////////////////////////////////
CRect CGraph::CalcDataArea()
{
	CRect dataarea;
	dataarea.left=m_iGraphX+CalcLeftMargin();
	dataarea.right=m_iGraphX+m_iGraphWidth-CalcRightMargin();
	dataarea.top=m_iGraphY+CalcTopMargin();
	dataarea.bottom=m_iGraphY+m_iGraphHeight-CalcBottomMargin();
	return dataarea;
}

////////////////////// CalcXAxisGridAndTicks ////////////////////////////////
/*
	This calculates the positions of the vertical gridlines of the plot area.
	This is also used for x-axis ticks as the ticks line up with the gridlines.
	How many they are is determined by the G_X_NUMTICKSANDGRID define in the header file
	Return: a double denoting the x-axis grid spacing
*/
////////////////////////////////////////////////////////////////////////////
double CGraph::CalcXAxisGridAndTicks()
{
	//the placing of the ticks co-incide with gridlines
	CRect rect=CalcDataArea();
	return ((double)rect.right-(double)rect.left)/(double)G_X_NUMTICKSANDGRID;

}

//////////////////  CalcYAxisGridAndTicks  ///////////////////////////////////
/*
	This calculates the positions of the horizontal gridlines of the plot area.
	This is also used for y-axis ticks as the ticks line up with the gridlines.
	How many they are is determined by the G_Y_NUMTICKSANDGRID  define in the header file
	Return:  A double denoting the horizontal gridline spacing
*/
////////////////////////////////////////////////////////////////////////////
double CGraph::CalcYAxisGridAndTicks()
```

```
{
	//the placing of the ticks co-incide with gridlines
	CRect rect=CalcDataArea();
	return ((double)rect.bottom-(double)rect.top)/(double)G_Y_NUMTICKSANDGRID;
}


//////////////////// DrawGraphTitle ////////////////////////////////////
/*
	This draws the Graph title string in the color m_crGraphTitleColor
	This title is centred abobe the plotting area.
*/
////////////////////////////////////////////////////////////////////////
void CGraph::DrawGraphTitle()
{
	//The graph title is drawn one character line down
	//centered left right between the left and right margins
	if (m_pWnd==NULL)
	{
		return;
	}
	UINT lmargin=CalcLeftMargin();
	UINT rmargin=CalcRightMargin();
	CRect rect;
	rect.left=m_iGraphX+lmargin;
	rect.top=m_iGraphY+m_iCharHeight;
	rect.right=m_iGraphX+m_iGraphWidth-rmargin;
	rect.bottom=rect.top+m_iCharHeight;

	//draw the title using the specified colorscheme
	//using the graph font
	CDC *pdc=m_pWnd->GetDC();
	CFont *poldfont;

	//note we must clear of any old stuff crap fom this area
	CBrush brush;
```

```
	brush.CreateSolidBrush(m_crGraphBkColor);
	//or bottom of a given rect
	pdc->FillRect(&rect,&brush);

	pdc->SetBkMode(TRANSPARENT);
	pdc->SetTextColor(m_crGraphTitleColor);
	poldfont=pdc->SelectObject(&m_GraphFont);
	pdc->DrawText(m_szGraphTitle,&rect,DT_CENTER|DT_END_ELLIPSIS);

	//cleanup
	pdc->SelectObject(poldfont);
	m_pWnd->ReleaseDC(pdc);
}

////////////////////// DrawXLegend ////////////////////////////////////////////////
/*
	The Xaxis legend is drawn below the plotting area below the x-axis scale numbers
*/
////////////////////////////////////////////////////////////////////////////////
void CGraph::DrawXLegend()
{
	//The x legend is drawn  below the data area
	if (m_pWnd==NULL)
	{
		return;
	}
	UINT lmargin=CalcLeftMargin();
	UINT rmargin=CalcRightMargin();
	CRect rect;
	rect.left=m_iGraphX+lmargin;
	rect.right=m_iGraphX+m_iGraphWidth-rmargin;
	rect.top=m_iGraphY+m_iGraphHeight-2*m_iCharHeight+1;
	rect.bottom=rect.top+m_iCharHeight;
	//just 2 b safe move the rect down a bit so we don't
	//interfere with the yaxis numbers
	rect.top+=1;
```

```
	rect.bottom+=1;

	//draw the title using the specified colorscheme
	//using the graph font
	CDC *pdc=m_pWnd->GetDC();
	CFont *poldfont;

	//note we must clear of any old stuff crap fom this area
	CBrush brush;
	brush.CreateSolidBrush(m_crGraphBkColor);
	rect.InflateRect(0,0,1,1);//bcause fillrect does not go right to the right
	//or bottom of a given rect
	pdc->FillRect(&rect,&brush);
	rect.DeflateRect(0,0,1,1);

	pdc->SetBkMode(TRANSPARENT);
	pdc->SetTextColor(m_crXLegendTextColor);
	poldfont=pdc->SelectObject(&m_GraphFont);
	pdc->DrawText(m_szXLegendText,&rect,DT_CENTER|DT_END_ELLIPSIS);

	//cleanup
	pdc->SelectObject(poldfont);
	m_pWnd->ReleaseDC(pdc);
}


////////////////////// DrawXAxisNumbers ////////////////////////////////////
/*
	The X axis scale numbers are drawm directly below the plot area.
	Three numbers are drawn, min, middle and max scaling
*/
//////////////////////////////////////////////////////////////////////////
void CGraph::DrawXAxisNumbers()
{
	//we draw three sets of numbers xmin, xmiddle and xmax
	//we will limit them to 7 digits
```

```
//
if (!m_pWnd)
{
        return;
}
CString astring;
CRect rect,dataarea;
CFont *poldfont;
CDC *pdc;

pdc=m_pWnd->GetDC();
poldfont=pdc->SelectObject(&m_GraphFont);
pdc->SetTextColor(m_crXLegendTextColor);
pdc->SetBkMode(TRANSPARENT);

dataarea=CalcDataArea();
//do the left side (min) left aligned
rect.left=dataarea.left;
rect.top=m_iGraphY+m_iGraphHeight-3*m_iCharHeight;
rect.right=rect.left+8*m_iCharWidth;
rect.bottom=rect.top+m_iCharHeight;
rect.top+=1;
rect.bottom+=1;
//clear any old text
CBrush brush;
brush.CreateSolidBrush(m_crGraphBkColor);
pdc->FillRect(&rect,&brush);
//format and print the number
astring.Format((CString)"%.4g",m_dXAxisMin);
pdc->DrawText(astring,&rect,DT_NOCLIP|DT_LEFT);
//do the half way point - centre aligned
rect.left=dataarea.left+((dataarea.right-dataarea.left)/2);
rect.left=rect.left-4*m_iCharWidth;
rect.right=rect.left+8*m_iCharWidth;
rect.top+=1;
rect.bottom+=1;
```

```
		pdc->FillRect(&rect,&brush);
		astring.Format((CString)"%.4g",(m_dXAxisMin+m_dXAxisMax)/2);
		pdc->DrawText(astring,&rect,DT_NOCLIP|DT_CENTER);
		//now do the righthand side (max); right aligned
		rect.left=dataarea.right-8*m_iCharWidth;
		rect.right=dataarea.right;
		rect.top+=1;
		rect.bottom+=1;;
		pdc->FillRect(&rect,&brush);
		astring.Format((CString)"%.4g",m_dXAxisMax);
		pdc->DrawText(astring,&rect,DT_NOCLIP|DT_RIGHT);

		//cleanup
		pdc->SelectObject(poldfont);
		m_pWnd->ReleaseDC(pdc);


}

////////////////////// DrawYAxisNumbers ////////////////////////////////////
/*
		The Y axis scale is drawn on the LHS of the plot area.
		Only two numbers are drawn - min and max to allow for the Y-axis legend.
*/
////////////////////////////////////////////////////////////////////////////
void CGraph::DrawYAxisNumbers()
{
		//we will only do two sets of numbers min and max because
		//if we print the halfway point it will cross the yaxis label
			if (!m_pWnd)
		{
			return;
		}
		CString astring;
		CRect rect,dataarea;
		CFont *poldfont;
```

```
CDC *pdc;

pdc=m_pWnd->GetDC();
poldfont=pdc->SelectObject(&m_GraphFont);
pdc->SetTextColor(m_crYLegendTextColor);
pdc->SetBkMode(TRANSPARENT);

dataarea=CalcDataArea();

//do the max value
rect.left=m_iGraphX-9*m_iCharWidth;
rect.top=dataarea.top-m_iCharWidth;
rect.right=rect.left+12*m_iCharWidth;//allow for  characters
rect.bottom=rect.top+m_iCharHeight;
//clear any old text
CBrush brush;
brush.CreateSolidBrush(m_crGraphBkColor);
//format and print the number
astring.Format((CString)"%.4g",m_dYAxisMax);
pdc->DrawText(astring,&rect,DT_RIGHT);

//do the 4/5 value
rect.top=dataarea.top+2*m_iCharHeight;
rect.bottom=rect.top+m_iCharHeight;
astring.Format((CString)"%.4g",(0.8*m_dYAxisMax));
pdc->DrawText(astring,&rect,DT_RIGHT);

//do the 3/5 value
rect.top=dataarea.top+4.5*m_iCharHeight;
rect.bottom=rect.top+m_iCharHeight;
astring.Format((CString)"%.4g",(0.6*m_dYAxisMax));
pdc->DrawText(astring,&rect,DT_RIGHT);

//do the 2/5 value
rect.top=dataarea.top+7*m_iCharHeight;
rect.bottom=rect.top+m_iCharHeight;
```

```
		astring.Format((CString)"%.4g",(0.4*m_dYAxisMax));
		pdc->DrawText(astring,&rect,DT_RIGHT);

		//do the 1/5 value
		rect.top=dataarea.top+9.5*m_iCharHeight;
		rect.bottom=rect.top+m_iCharHeight;
		astring.Format((CString)"%.4g",(0.2*m_dYAxisMax));
		pdc->DrawText(astring,&rect,DT_RIGHT);

		//do the bottom - (min)
		rect.top=dataarea.bottom-0.5*m_iCharHeight;
		rect.bottom=rect.top+m_iCharHeight;
		astring.Format((CString)"%.4g",(m_dYAxisMin));
		pdc->DrawText(astring,&rect,DT_RIGHT);

		//cleanup
		pdc->SelectObject(poldfont);
		m_pWnd->ReleaseDC(pdc);
}


////////////////// SetdefaultColorScheme  ////////////////////////////////
/*
		sets the colours of various bits back to the default colours
*/
//////////////////////////////////////////////////////////////////////////
void CGraph::SetDefaultColorScheme()
{
		SetColorScheme(G_DEFAULTSCHEME);
}


//////////////////////  SetDefaultGraphTitle ///////////////////////////////
/*
		Sets the default graph title
*/
```

```
////////////////////////////////////////////////////////////////////////
void CGraph::SetDefaultGraphTitle()
{
	m_szGraphTitle="Variation in DOP over 24hr Period";
}


////////////// SetDefaultXLegend//////////////////////////////////////////////
/*

*/
////////////////////////////////////////////////////////////////////////
void CGraph::SetDefaultXLegend()
{
	m_szXLegendText="Time in 24hr Format";
}


////////////////////////// SetGraphTitle ////////////////////////////////////
/*
	This sets the GraphTitle
	Takes:
	CString
*/
////////////////////////////////////////////////////////////////////////
void CGraph::SetGraphTitle(CString GraphTitle)
{
	m_szGraphTitle=GraphTitle;
	DrawGraphTitle();
}


////////////////////////// SetXLegendText  ///////////////////////////////////
/*
	Sets the X axis legend text
*/
```

```
////////////////////////////////////////////////////////////////////////
void CGraph::SetXLegendText(CString XText)
{
	m_szXLegendText=XText;
	DrawXLegend();
}


////////////////////// SetYLineAtLeft ///////////////////////////////////////
/*
	The Y axis line can be forced to the LHS of the plot area using this function
*/
////////////////////////////////////////////////////////////////////////////

void CGraph::SetYLineAtLeft(BOOL AtLeft)
{
	BOOL bprevious=m_bYLineAtLeft;
	m_bYLineAtLeft=AtLeft;
	//if there is a change in the Y line position then we will have to
	//redraw the graph
	if (m_bYLineAtLeft != bprevious)
	{
		PaintGraph();
	}
}

////////////////////////// ShowGrid /////////////////////////////////////////
/*
	This switches the grid on or off as set by the bShow parameter
	The graph is repainted to match
*/
////////////////////////////////////////////////////////////////////////////
void CGraph::ShowGrid(BOOL bShow)
{
	//this is public function
	//show the graph grid if bShow==TRUE
```

```
	//or vice-versa
	BOOL bprevious=m_bShowGrid;
	m_bShowGrid=bShow;
	//if there is a change then repaint
	if (m_bShowGrid !=bprevious)
	{
		PaintGraph();
	}
}

////////////////////// ShowTicks ////////////////////////////////////////////
/*
	This switches the x and y axis 'ticks' on or off as set by the
	bShow parameter
*/
//////////////////////////////////////////////////////////////////////////
void CGraph::ShowTicks(BOOL bShow)
{
	//this is public function
	//show the graph grid if bShow==TRUE
	//or vice-versa
	BOOL bprevious=m_bShowTicks;
	m_bShowTicks=bShow;
	//if there is a change then repaint
	if (m_bShowTicks!=bprevious)
	{
		PaintGraph();
	}

}

////////////////////// ConvertToGraphCoords ////////////////////////////////////
/*
```

ConvertToGraphCoords (double x, double y) will take the result of some calculation as given by x and y and return where they should be plotted on the graph.

As we are can only plot a whole pixel the return value is of type LONG (as apposed to double).

The Y-pixel is in the HIWORD and x-pixel in the LOWORD of the return.

Note that depending on the scale of the graph, the return cords of a single pixel could be one of many.

For example: if the x axis is 400 pixels wide, but is scaled to represent 1000, then each pixel represent 2.5 in the real world.

So to pixel 398 represent the real world values of 995 to 996.5 inc.

*/

```
////////////////////////////////////////////////////////////////////////////
LONG CGraph::ConvertToGraphCoords(double x, double y)
{

	LONG result =-1;
	//to be plottable on the graph the given x-value must be between
	//x-min and x-max
	if(x < m_dXAxisMin || x > m_dXAxisMax)
	{
		return result;
	}
	if( y <m_dYAxisMin || y > m_dYAxisMax)
	{
		return result;
	}

	//calc the abs difference between Xmin and x;
	double xdif = abs(m_dXAxisMin - x);
	//calc the abs difference between Ymin and y;
	double ydif = abs(m_dYAxisMin - y);

	//find the dataarea
	CRect rect=CalcDataArea();

	int xpos=rect.left+(xdif*m_dPixelsPerX); //from left
```

```
	int ypos=rect.bottom-(ydif*m_dPixelsPerY); //from bottom

	result=MAKELONG(xpos,ypos);

	return result;
}


////////////////////////////// DoFunction ////////////////////////////////////
/*

	The user fills in a G_FUNCTIONSTRUCT relevant to a function and passes
	a pointer to it to this function.

	The pointer to this G_FUNCTIONSTRUCT is saved in a member variable
	This function does some preliminary stuff  and if there are no
	obvious problems it then calls the PaintGraph function.


	Returns:

	FALSE if there are no problems

*/

//////////////////////////////////////////////////////////////////////////////
BOOL CGraph::DoFunction(G_FUNCTIONSTRUCT *pFunctionParams)
{
	//do some checks first to see if good data has been passed
	if( (pFunctionParams->pPlotXYItems==NULL) && (pFunctionParams-
>FuncType==G_PLOTXY))
	{
		return FALSE;
	}

	m_pFunctionParams=pFunctionParams;
	//Set chart title and other text items
```

```
	SetGraphTitle((CString)m_pFunctionParams->szGraphTitle);


	//////////////////AutoScaling X axis
	/*
	Precautions:
	For the plotdeviationpercent and plotdeviationabsolute
	the axis min should be 0 (having any other values makes no sense)
	*/
	double xminimum;

	xminimum=pFunctionParams->xMin;

	SetXAxisScale(xminimum,pFunctionParams->xMax);
	//set the Y axis scale
	SetYAxisScale(pFunctionParams->yMin,pFunctionParams->yMax);

	PaintGraph();
	return TRUE;
}


////////////////////////// ClearFunction ////////////////////////////////
/*

	This resets the G_FUNCTIONSTRUCT member pointer
	The graph is repainteed (cleared)
*/

//////////////////////////////////////////////////////////////////////////
void CGraph::ClearFunction()
{
	m_pFunctionParams=NULL; //reset the pointer
	//Clear the graph
	PaintGraph();
}
```

```
////////////////////////////// DrawFunction  ////////////////////////////////
/*

//        This is called from within the PaintGraph routine to draw
//        the actual function onto the graph.

//        This functions just switches the FunctionType member specified in the
//        G_FUNCTIONSTRUCT and calls the appropriate routine.

*/

//////////////////////////////////////////////////////////////////////////////
void CGraph::DrawFunction()
{
        if(!m_pWnd)
        {
                return;
        }

        if(!m_pFunctionParams)
        {
                return;
        }

        switch (m_pFunctionParams->FuncType)
        {
                case G_PLOTXY:
                        DoPlotXY();
                        break;
                default:
                break;
        }
}

//////////////////////  DoPlotXY ////////////////////////////////////////////////
/*
```

```
        This has many similarities with the other functions - however
        there is no need to calculate Y  in the sane way as in the other function
        as it is given.

        a particular x point on the graph is also given - we place the
        given y value at the given x point

        Other:

        a. We will not bother plotting  if x < the graph scale XMin or > XMax
        b. Only dot, bar and line will be acceptable for the chart type. Any other
        will default to bar. However be aware that line is only suitable if the
        x-values are in sequence.

        c. The usual y constraints apply

*/


////////////////////////////////////////////////////////////////////////////
void CGraph::DoPlotXY()
{
        UINT prevx=0;
        UINT prevy=0;
        BOOL firstpoint=TRUE;

        LONG result;
        UINT xstart;
        UINT ystart;
        double xperpixel= 1/m_dPixelsPerX;
        double yperpixel=1/m_dPixelsPerY;

        result=ConvertToGraphCoords(m_dXAxisMin,m_dYAxisMin);
        xstart=LOWORD(result);//the left hand side of the graph plot area on screen
        ystart=HIWORD(result);//shouldbe the bottom of the graph plot area on screen

        for (UINT count =0; count < m_pFunctionParams->num_PlotXYItems*2; count+=2)
        {
```

```
                double x=m_pFunctionParams->pPlotXYItems[count];
                double y=m_pFunctionParams->pPlotXYItems[count+1];

                //if x is off scale - don't bother
                        if ( (x < m_dXAxisMin) || (x > m_dXAxisMax))
                        {
                                continue; //NEXT !!!!!!
                        }

                UINT pixelx= xstart+(x-m_dXAxisMin)/xperpixel;

                //y=ConstrainY(y);
                UINT pixely=ystart-(y-m_dYAxisMin)/yperpixel;

                if(firstpoint)
                {
                        prevx=pixelx;
                        prevy=pixely;
                        firstpoint=FALSE;
                }
                PlotPoints(pixelx,pixely,prevx,prevy);
                //current point becomes previous point
                prevx=pixelx;
                prevy=pixely;

        }
}


////////////////// PlotPoints  ////////////////////////////////////////////
/*

        Each Function e.g. DoSineX,DoPlotXY, etc, calls this function as
        they calculate each point so that each point can be drawn on the
        Takes:
```

```
        UINT x, UNIT y - the graph co-ord of the point just calculeted
        (current point).

        UINT prevx, UINT prevy - the co-ords of the previous point
        This routine checks what type of plot (line, dot, or bar) is
        required and calls the appropriate routine
*/

////////////////////////////////////////////////////////////////////////////
void CGraph::PlotPoints(UINT x, UINT y, UINT prevx, UINT prevy)
{
        //here we check the chart type and plot the points accordingly
        //we need to constarin the Y values to keep them within the
        //plot area;

        switch(m_pFunctionParams->ChartType)
        {
                case G_LINECHART:
                {
                        DrawConnectLine(prevx,prevy,x,y);
                        break;
                }
        }//SWITCH

                return;
}


//////////////////////////// DrawConnectLine ////////////////////////////////
/*

        For the line chat type, this routine draws a line between previous
        point (FROM) and current point (TO)

*/

////////////////////////////////////////////////////////////////////////////
void CGraph::DrawConnectLine(UINT FromX, UINT FromY, UINT ToX, UINT ToY)
```

```
{
	//draws a connecting line between to pixels
	//using the graphpen color

	if(!m_pWnd)
	{
		return;
	}

	CPen pen, *poldpen;
	pen.CreatePen(PS_SOLID,1,m_crGraphPenColor);
	CDC *pdc=m_pWnd->GetDC();
	poldpen=pdc->SelectObject(&pen);
	pdc->MoveTo(FromX,FromY);
	pdc->LineTo(ToX,ToY);
	pdc->SelectObject(poldpen);
	m_pWnd->ReleaseDC(pdc);
}
```

## D. GRAPH.H [AFTER 20]

```
// Graph.h: interface for the CGraph class.
//
//////////////////////////////////////////////////////////////////////

#if !defined(AFX_GRAPH_H__70FB8DF3_88AC_40C5_802C_58621127B9E9__INCLUDED_)
#define AFX_GRAPH_H__70FB8DF3_88AC_40C5_802C_58621127B9E9__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000

// Some  defines

//colorschemes
#define G_DEFAULTSCHEME 0
#define G_WHITESCHEME 1

//default graphsize
#define G_MINGRAPHWIDTH 300
#define G_MINGRAPHHEIGHT 200

//default axies scaling
#define G_DEFAULTXMIN 0
#define G_DEFAULTXMAX 24
#define G_DEFAULTYMIN 0
#define G_DEFAULTYMAX 5

//miscellaneous
#define G_X_NUMTICKSANDGRID 12 //how may parts the dataarea is divided
#define G_Y_NUMTICKSANDGRID 5 //how may parts the dataarea is divided
#define G_TICKLENGTH 10 //size of those little ticks on the axes

/*
Function related  defines and stuff
```

```
*/
//for the builtin functions
#define NUMFUNCTIONS 1
//Plot function
#define G_PLOTXY         1

//plot type bar,line, etc...
#define NUMCHARTTYPES 1
#define G_LINECHART 1 // each point is a line drawn from each x-y point to the next

//some structures for passing data
typedef struct
{
        UINT FuncType;
        UINT ChartType;//line
        double xMin;
        double xMax;
        double yMin;
        double yMax;
        char   *szGraphTitle;
        char       *szXLegend;
        double *pPlotXYItems;
        int    num_PlotXYItems;

}G_FUNCTIONSTRUCT, *LPG_FUNCTIONSTRUCT;


class CGraph
{
public:
        void ShowTicks(BOOL bShow);
        void ClearFunction(void);
        BOOL DoFunction(G_FUNCTIONSTRUCT *pFunctionParams);
        void ShowGrid(BOOL bShow);
        void SetYLineAtLeft(BOOL AtLeft);
        void GraphSetAllDefaults();
```

```
        void SetXLegendText(CString XText);
        void SetGraphTitle(CString GraphTitle);
        void PaintGraph(void);
        CGraph(CWnd *pParentWnd,int xPos=0, int yPos=0, int Width =0, int Height=0, UINT
colorscheme=G_DEFAULTSCHEME);
        void SetYAxisScale(double min, double max);
        void SetXAxisScale(double min,double max);
        void SetGraphSizePos(int xPos, int yPos, int Width, int Height);
        void SetColorScheme(int Scheme, BOOL bRedraw=FALSE);
        void CreateGraphFont(CString FaceName,UINT size);
        CGraph();
        virtual ~CGraph();



private:
        LONG ConvertToGraphCoords(double x, double y);
        void DoPlotXY();
        void DrawConnectLine(UINT FromX, UINT FromY, UINT ToX, UINT ToY);
        void PlotPoints(UINT x, UINT y, UINT prevx, UINT prevy);
        void DrawFunction();
        void SetDefaultColorScheme(void);
        CString m_szFunctionNameText;
        COLORREF m_crFunctionNameColor;
        void DrawYAxisNumbers();
        void DrawXAxisNumbers(void);
        void DrawTicks(void);
        double CalcYAxisGridAndTicks(void);
        CRect CalcDataArea(void);
        double CalcXAxisGridAndTicks(void);
        void DrawGrid(void);
        void DrawXLegend();
        void SetDefaultXLegend(void);
        void SetDefaultGraphTitle(void);
        void DrawGraphTitle();
        UINT CalcRightMargin();
        UINT CalcLeftMargin();
```

```
UINT CalcBottomMargin();
UINT CalcTopMargin();
BOOL m_bShowTicks;//Ticks are the little things on the x & y axis
BOOL m_bShowGrid;
BOOL m_bAutofit;
BOOL m_bYLineAtLeft;
CString m_szFontFace;
CString m_szXLegendText;
CString m_szGraphTitle;
COLORREF m_crYTickColor;
COLORREF m_crXTickColor;
COLORREF m_crYLegendTextColor;
COLORREF m_crXLegendTextColor;
COLORREF m_crGraphTitleColor;
COLORREF m_crGraphPenColor;
COLORREF m_crGraphBkColor;
COLORREF m_crGridColor;
int m_iFontSize;//
int m_iGraphWidth;//
int m_iGraphHeight;//
int m_iGraphX;//location of the fraph within the window
int m_iGraphY;//location of the graph within the window
double m_dXAxisMin;// the start value of X
double m_dYAxisMin;//start value of Y
double m_dXAxisMax;
double m_dYAxisMax;
CWnd *m_pWnd;//parent/owner
//Helper calculated values - that is to say that these
//values are not passed in to the graph
//they are calculated from other given
CFont m_GraphFont;//default font font created from default fontface, & point size
int m_iCharHeight;//calculated from the font
int m_iCharWidth;//calculated from the font
int m_iOriginX;//location of the origin within the graph
int m_iOriginY;//location of the origin within the graph
double m_dPixelsPerY;//scaling
```

```
        double m_dPixelsPerX;//scaling
        int m_iScrollPosX;//
        int m_iScrollPosY;//
        //Data related variables
        G_FUNCTIONSTRUCT *m_pFunctionParams;
};

#endif // !defined(AFX_GRAPH_H__70FB8DF3_88AC_40C5_802C_58621127B9E9__INCLUDED_)
```

## E. MATRIX.H [AFTER 21]

```
#include <assert.h> // Defines the assert function.

class Matrix {

public:

// Default Constructor. Creates a 1 by 1 matrix; sets value to zero.
Matrix () {
  nRow_ = 1; nCol_ = 1;
  data_ = new double [1];  // Allocate memory
  set(0.0);                // Set value of data_[0] to 0.0
}

// Regular Constructor. Creates an nR by nC matrix; sets values to zero.
// If number of columns is not specified, it is set to 1.
Matrix(int nR, int nC = 1) {
  assert(nR > 0 && nC > 0);    // Check that nC and nR both > 0.
  nRow_ = nR; nCol_ = nC;
  data_ = new double [nR*nC];  // Allocate memory
  assert(data_ != 0);          // Check that memory was allocated
  set(0.0);                    // Set values of data_[] to 0.0
}

// Copy Constructor.
// Used when a copy of an object is produced
// (e.g., passing to a function by value)
Matrix(const Matrix& mat) {
  this->copy(mat);   // Call private copy function.
}

// Destructor. Called when a Matrix object goes out of scope.
~Matrix() {
  delete [] data_;   // Release allocated memory
}
```

```
// Assignment operator function.
// Overloads the equal sign operator to work with
// Matrix objects.
Matrix& operator=(const Matrix& mat) {
  if( this == &mat ) return *this;  // If two sides equal, do nothing.
  delete [] data_;                  // Delete data on left hand side
  this->copy(mat);                  // Copy right hand side to l.h.s.
  return *this;
}

// Simple "get" functions. Return number of rows or columns.
int nRow() const { return nRow_; }
int nCol() const { return nCol_; }

// Parenthesis operator function.
// Allows access to values of Matrix via (i,j) pair.
// Example: a(1,1) = 2*b(2,3);
// If column is unspecified, take as 1.
double& operator() (int i, int j = 1) {
  assert(i > 0 && i <= nRow_);          // Bounds checking for rows
  assert(j > 0 && j <= nCol_);          // Bounds checking for columns
  return data_[ nCol_*(i-1) + (j-1) ];  // Access appropriate value
}

// Parenthesis operator function (const version).
const double& operator() (int i, int j = 1) const{
  assert(i > 0 && i <= nRow_);          // Bounds checking for rows
  assert(j > 0 && j <= nCol_);          // Bounds checking for columns
  return data_[ nCol_*(i-1) + (j-1) ];  // Access appropriate value
}

// Set function. Sets all elements of a matrix to a given value.
void set(double value) {
  int i, iData = nRow_*nCol_;
  for( i=0; i<iData; i++ )
```

```
    data_[i] = value;
}

static double inv(Matrix A, Matrix& Ainv) {
// Compute inverse of matrix
// Input
//    A    -    Matrix A (N by N)
// Outputs
//    Ainv  -    Inverse of matrix A (N by N)
  int N = A.nRow();
  assert( N == A.nCol() );

  Ainv = A;  // Copy matrix to ensure Ainv is same size

  int i, j, k;
  Matrix scale(N), b(N,N);  // Scale factor and work array
  int *index;  index = new int [N+1];

  //* Matrix b is initialized to the identity matrix
  b.set(0.0);
  for( i=1; i<=N; i++ )
            b(i,i) = 1.0;

  //* Set scale factor, scale(i) = max( |a(i,j)| ), for each row
  for( i=1; i<=N; i++ ) {
            index[i] = i;                                  // Initialize row index list
            double scalemax = 0.;
            for( j=1; j<=N; j++ )
              scalemax = (scalemax > fabs(A(i,j))) ? scalemax : fabs(A(i,j));
            scale(i) = scalemax;
  }

  //* Loop over rows k = 1, ..., (N-1)
  int signDet = 1;
  for( k=1; k<=N-1; k++ ) {
            //* Select pivot row from max( |a(j,k)/s(j)| )
```

```
        double ratiomax = 0.0;
        int jPivot = k;
        for( i=k; i<=N; i++ ) {
          double ratio = fabs(A(index[i],k))/scale(index[i]);
          if( ratio > ratiomax ) {
                jPivot=i;
                ratiomax = ratio;
          }
        }
        //* Perform pivoting using row index list
        int indexJ = index[k];
        if( jPivot != k ) {          // Pivot
          indexJ = index[jPivot];
          index[jPivot] = index[k];   // Swap index jPivot and k
          index[k] = indexJ;
          signDet *= -1;                          // Flip sign of determinant
        }
        //* Perform forward elimination
        for( i=k+1; i<=N; i++ ) {
          double coeff = A(index[i],k)/A(indexJ,k);
          for( j=k+1; j<=N; j++ )
                A(index[i],j) -= coeff*A(indexJ,j);
          A(index[i],k) = coeff;
          for( j=1; j<=N; j++ )
                b(index[i],j) -= A(index[i],k)*b(indexJ,j);
        }
}
//* Compute determinant as product of diagonal elements
double determ = signDet;   // Sign of determinant
for( i=1; i<=N; i++ )
        determ *= A(index[i],i);

//* Perform backsubstitution
for( k=1; k<=N; k++ ) {
        Ainv(N,k) = b(index[N],k)/A(index[N],N);
        for( i=N-1; i>=1; i--) {
```

```
            double sum = b(index[i],k);
            for( j=i+1; j<=N; j++ )
                    sum -= A(index[i],j)*Ainv(j,k);
            Ainv(i,k) = sum/A(index[i],i);
          }
  }

  delete [] index;   // Release allocated memory
  return( determ );
}


//*********************************************************************
private:

// Matrix data.
int nRow_, nCol_;  // Number of rows, columns
double* data_;     // Pointer used to allocate memory for data.

// Private copy function.
// Copies values from one Matrix object to another.
void copy(const Matrix& mat) {
  nRow_ = mat.nRow_;
  nCol_ = mat.nCol_;
  int i, iData = nRow_*nCol_;
  data_ = new double [iData];
  for(i = 0; i<iData; i++ )
    data_[i] = mat.data_[i];
}

}; // Class Matrix
```

# APPENDIX C. MATLAB CODES FOR DOP CALCULATION

```
% Code to obtain DOP (ENU) from Almanac Data
% By Yuen Ming Fatt
% Last updated on 27 Feb 2009
%*********************************************************************
%*********************************************************************

clear all
clc
format long g

%*********************************************************************
%Convert Target's Latitude, Longitude and Altitude to ECEF Coordinates
%*********************************************************************
%Target's Latitude, Longitude and Altitude Input
lat_deg = 0; %Latitude (degree) (user input)#####
lon_deg = 90; %Longitude (degree) (user input)#####
alt = 0; %Altitude (meter) (user input)#####
sealevel = 0;
lat = lat_deg.*pi./180; %Latitude (rad)
lon = lon_deg.*pi./180; %Longitude (rad)
%-------------------------
%WGS84 ellipsoid constants
a = 6378137;
es = 8.1819190842622e-2;
%-------------------------
%Intermediate calculation
N = a./sqrt(1-es.^2.*sin(lat).^2); %Prime vertical radius of curvature
%Results
xTgt = (N+alt).*cos(lat).*cos(lon); %Target x ECEF coordinate (meter)
yTgt = (N+alt).*cos(lat).*sin(lon); %Target y ECEF coordinate (meter)
zTgt = ((1-es.^2).*N + alt).*sin(lat); %Target z ECEF coordinate
(meter)

xLocalRef = (N+sealevel).*cos(lat).*cos(lon); %ENU Local ref pt x ECEF
coordinate (meter)
yLocalRef = (N+sealevel).*cos(lat).*sin(lon); %ENU Local ref pt y ECEF
coordinate (meter)
zLocalRef = ((1-es.^2).*N + sealevel).*sin(lat); %ENU Local ref pt z
ECEF coordinate (meter)
%End of conversion


%*********************************************************************
%Convert Almanac Data to Satellite Position in ECEF Coordinates and
check the Line of Sight to Target
%*********************************************************************
%Constants
io = 0.3.*pi; %Inclination angle @ ref. time (rad)
mju = 3.986005e14; %WGS 84 value of the Earth's universal gravitational
parameter for GPS user (meters^3/sec^2)
```

```
OMEGAdote = 7.2921151467e-5; %WGS 84 value of the Earth's rotation rate
(rad/sec)

%-------------------------
%Almanac Data from Satellite
fid = fopen('current.al3'); %Open source file "current.al3"
%Common Data
NumSV = fscanf(fid, '%d'); %Number of Satellites
name = fgetl(fid);
[data, count] = fscanf(fid,'%f');
fclose(fid);
wn = data(1); %GPS week no.
toa = data(2); %Time of Applicability of Almanac(sec) (range: 0 to
604,784)

%-------------------------
%Input time from user#####
y = 2009; %Year
m = 2; %Month
d = 3; %Date
h = 0; %Hours
mi = 0; %Minutes
sec = 0; %Seconds
timezone = 0; %Timezone (Eastern Standard Time (North America) = -5hr)
summertime = 0; %To account for daylight saving. If summer, 1 = Yes, 0
= No

%-------------------------
[gps_week, sec_of_week] = ymdhms2gps(y, m, d, h, mi, sec, timezone,
summertime);
Total_weeks = gps_week; %Total number of weeks since 6 Jan 1980
while gps_week >= 1024
    gps_week = gps_week - 1024;
end
tk = (gps_week - wn)*604800 + (sec_of_week - toa); %Time since toa(sec)
(range: -302,400 to 302,400)
if gps_week < wn
    fprintf('Almanac file used is incorrect. Please use almanac file
for week %4.0f',gps_week)
elseif gps_week > wn
    fprintf('Almanac file is outdated. Please use almanac file for week
%4.0f',gps_week)
end

%-------------------------
%Satellite Specific Data
num = 0;
SVcount = 1;
while SVcount <= NumSV,
    PRN = data(num+3); %PRN number
    SVN = data(num+4); %Satllite number
    URA = data(num+5); % Average URA number
    ec = data(num+6); %Eccentricity (dimensionless) (range: 0-0.03)
    del_ik = data(num+7).*pi; %Inclination correction (rad)
    OMEGAdot = data(num+8).*pi; %Rate of right ascension (rad/sec)
```

```
    sqrtA = data(num+9); %Sqr root semi-major axis (m^1/2)
    OMEGAo = data(num+10).*pi; %Right ascension @ ref. time (rad)
    omega = data(num+11).*pi; %Argument of perigee (rad)
    Mo = data(num+12).*pi; %Mean Anomaly @ ref. time (rad)
    Af0 = data(num+13); %Clock offset (sec)
    Af1 = data(num+14); %Clock drift (sec/sec)
    Health = data(num+15); %Satellite Health; 0=healthy
    num = num+14;
    %End of data extraction
    %-------------------------
    %Calculations
    A = sqrtA.^2; %Orbit semi-major axis (meter)
    n = sqrt(mju./(A.^3)); %Computed mean motion (rad/sec)
    Mk = Mo+tk.*n; %Mean anomaly (rad)
    %Start values for iterative solution of Kepler eq.
    Ek = Mk;
    Eold=0;
    while abs(Ek-Eold)>=1.0e-10
        Eold = Ek;
        Ek = Mk+ec.*sin(Ek);
    end
    %End of iteration
    vk = atan2((sqrt(1-ec.^2).*sin(Ek))./(1-ec.*cos(Ek)),(cos(Ek)-
ec)./(1-ec.*cos(Ek))); %True anomaly (rad)
    Ek = acos((ec+cos(vk))./(1+ec.*cos(vk)));
    uk = omega+vk; %Argument of latitude (rad)
    rk = A.*(1-ec.*cos(Ek)); %Corrected radius (meter)
    ik = io+del_ik; %Corrected inclination (rad)
    xk1 = rk.*cos(uk); %x position in orbital plane (meter)
    yk1 = rk.*sin(uk); %y position in orbital plane (meter)
    OMEGAk = OMEGAo+(OMEGAdot-OMEGAdote).*tk-OMEGAdote.*(toa);
%Corrected longitude of ascending node (rad)
    %-------------------------
    %Calculations for ECEF coordinates
    xk(SVcount) = xk1.*cos(OMEGAk)-yk1.*cos(ik).*sin(OMEGAk);
%Satellite x ECEF coordinate (meter)
    yk(SVcount) = xk1.*sin(OMEGAk)+yk1.*cos(ik).*cos(OMEGAk);
%Satellite y ECEF coordinate (meter)
    zk(SVcount) = yk1.*sin(ik); %Satellite z ECEF coordinate (meter)
    %End of ECEF coordinates conversion from Almanac Data
    %-------------------------
    %Covert ECEF coordinates to East-North-Up Coordinates
    East = -sin(lon).*(xk-xLocalRef) + cos(lon).*(yk-yLocalRef);
    North = -sin(lat).*cos(lon).*(xk-xLocalRef) -
sin(lat).*sin(lon).*(yk-yLocalRef) + cos(lat).*(zk-zLocalRef);
    Up = cos(lat).*cos(lon).*(xk-xLocalRef) + cos(lat).*sin(lon).*(yk-
yLocalRef) + sin(lat).*(zk-zLocalRef);
    %-------------------------
    %Algorithm to determine Line of Sight between Target and Satellite
    Obstruction = 10; %Obstruction to the Field of View (deg) (user
input)#####
    mag_Tgt = sqrt(xTgt.^2+yTgt.^2+zTgt.^2); %Distance of Target from
Earth center (meter)
    xTgttoSV(SVcount) = xk(SVcount) - xTgt;
    yTgttoSV(SVcount) = yk(SVcount) - yTgt;
```

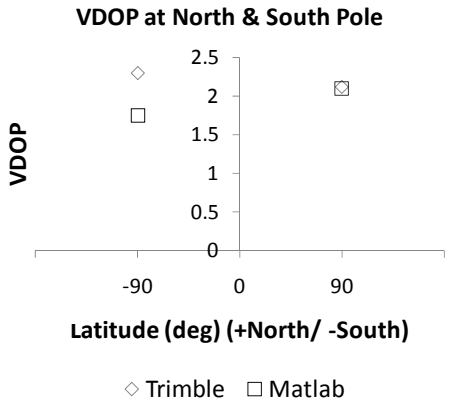
```
    zTgttoSV(SVcount) = zk(SVcount) - zTgt;
    mag_TgttoSV(SVcount) =
sqrt((xTgttoSV(SVcount)).^2+(yTgttoSV(SVcount)).^2+(zTgttoSV(SVcount)).
^2); %Distance from Target to Satellite
    AngleFromTgt(SVcount) = acos(((xTgttoSV(SVcount).*xTgt) +
(yTgttoSV(SVcount).*yTgt) +
(zTgttoSV(SVcount).*zTgt))./(mag_TgttoSV(SVcount).*mag_Tgt));

    %if mag_SVproj(SVcount)>mag_Tgt && AngleTOS(SVcount)<(pi/2) &&
AngleFromTgt(SVcount)<(pi/2-(Obstruction*pi/180)) && Health ==0
    if AngleFromTgt(SVcount)<(pi/2-(Obstruction*pi/180)) && Health ==0
        Los = 1; %There is Line of Sight
    else
        Los = 0; %There is NO Line of Sight
    end
    LOS(SVcount) = Los;
    SVcount = SVcount+1;
end


%**************************************************************************
%Assign coordinates to Target and valid Satellites (i.e. those with LOS
with Target)
%**************************************************************************
num3 = 1;
num4 = 1;
while num3 <= NumSV
    if LOS(num3) == 1
    SV(num4,1:3) = [East(num3), North(num3), Up(num3)]; %Assigning
coordinates to respective valid satellites
    num4 = num4+1;
    end
    num3 = num3+1;
end
NumValidSV = num4-1;


%**************************************************************************
%Calculate PDOP with the valid satellites i.e. those within the LOS of
Tgt
%**************************************************************************
%Pseudo-Range and Directional Derivative Loop
for num4 = 1:NumValidSV
     %Calculate pseudo-ranges from reciever position to other vehicles
     r(num4) = sqrt((SV(num4,1))^2 + (SV(num4,2))^2 + (SV(num4,3)-
alt)^2);
     %Calculate directional derivatives for X,Y,Z, and Time
     Dx(num4) = (SV(num4,1)-0)/r(num4); %x-coordinates of Tgt in ENU
frame is zero
     Dy(num4) = (SV(num4,2)-0)/r(num4); %y-coordinates of Tgt in ENU
frame is zero
     Dz(num4) = (SV(num4,3)-alt)/r(num4); %z-coordinates of Tgt in ENU
frame is the altitude
     Dt(num4) = -1;
```

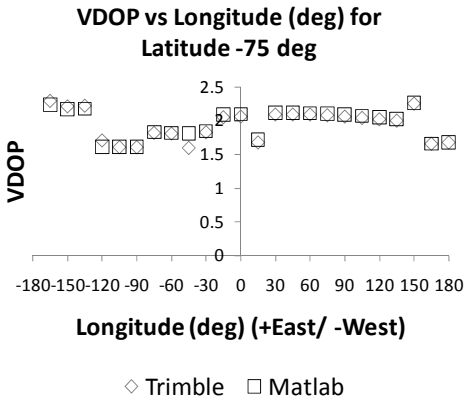
```
end

%Produce the Covariance Matrix from the Directional Derivatives
Alp = zeros(NumValidSV,4);
for num5 = 1:NumValidSV
    Alp(num5,1) = Dx(num5);
    Alp(num5,2) = Dy(num5);
    Alp(num5,3) = Dz(num5);
    Alp(num5,4) = Dt(num5);
end
Brv = transpose(Alp);
Chl = Brv*Alp;
Dlt = inv(Chl);

% Calculate DOPs from the diagonal elements of the Covariance Matrix
GDOP = sqrt(Dlt(1,1) + Dlt(2,2) + Dlt(3,3) + Dlt(4,4));
PDOP = sqrt(Dlt(1,1) + Dlt(2,2) + Dlt(3,3));
HDOP = sqrt(Dlt(1,1) + Dlt(2,2));
TDOP = sqrt(Dlt(4,4));
VDOP = sqrt(Dlt(3,3));
YDOP = sqrt(Dlt(2,2));
XDOP = sqrt(Dlt(1,1));

Results(1) = NumValidSV;
Results(2) = GDOP;
Results(3) = PDOP;
Results(4) = HDOP;
Results(5) = TDOP;
Results(6) = VDOP;
Results(7) = YDOP;
Results(8) = XDOP;

%End of Code :)
```

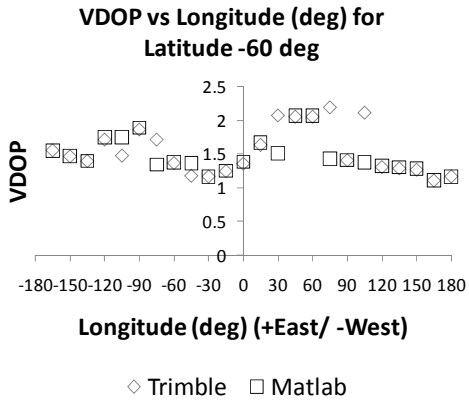
THIS PAGE INTENTIONALLY LEFT BLANK

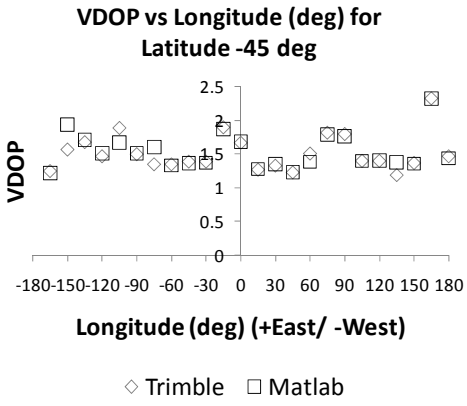
# APPENDIX D. DOP COMPARISON BETWEEN MATLAB & TRIMBLE

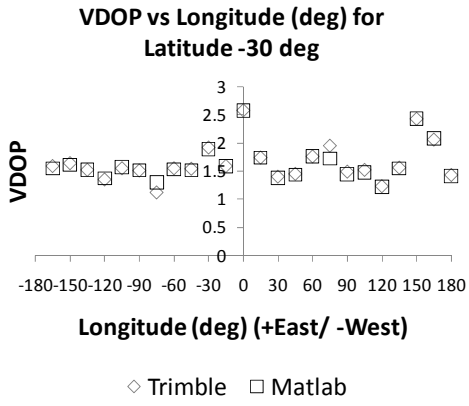
## A. VDOP

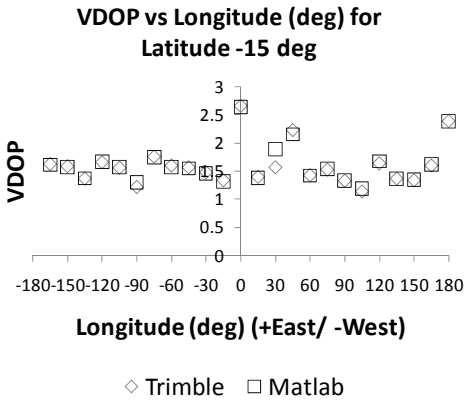


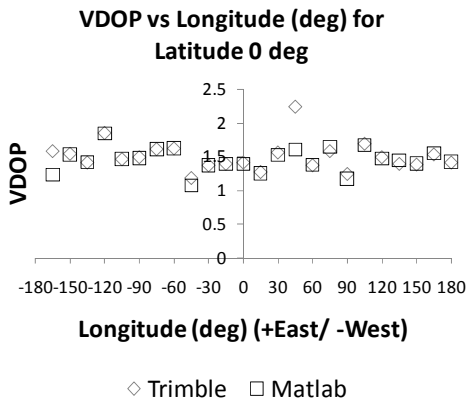



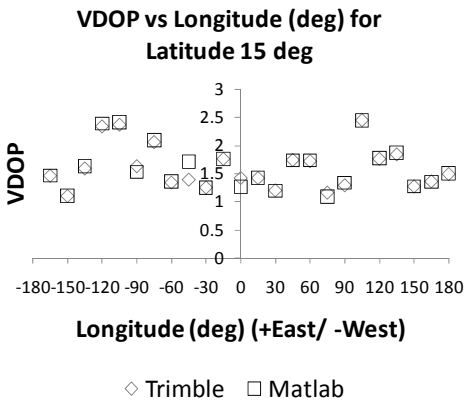



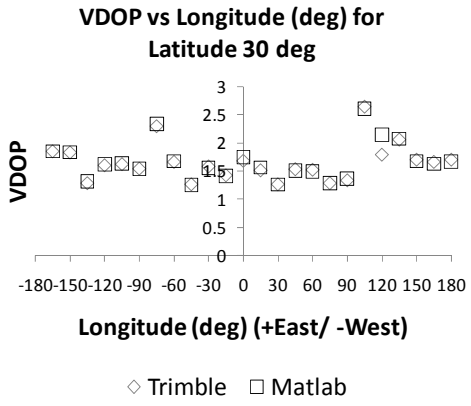



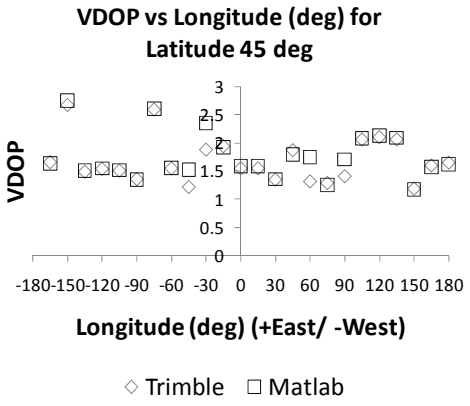



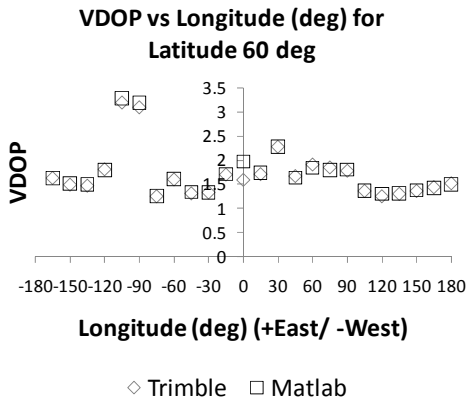



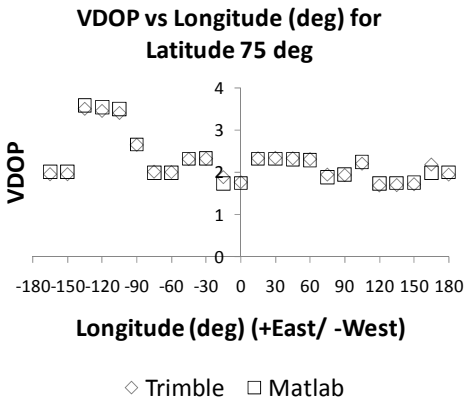

VDOP vs Longitude (deg) for Latitude 0 deg
VDOP
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab


VDOP vs Longitude (deg) for Latitude 15 deg
VDOP
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

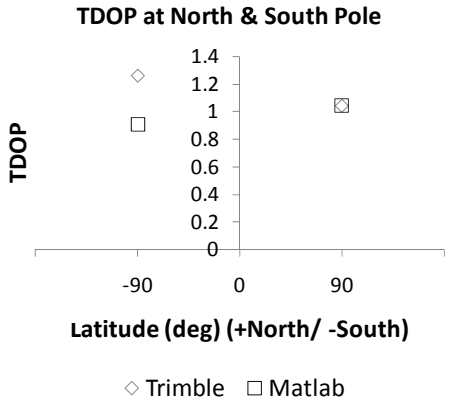

VDOP vs Longitude (deg) for Latitude 30 deg
VDOP
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

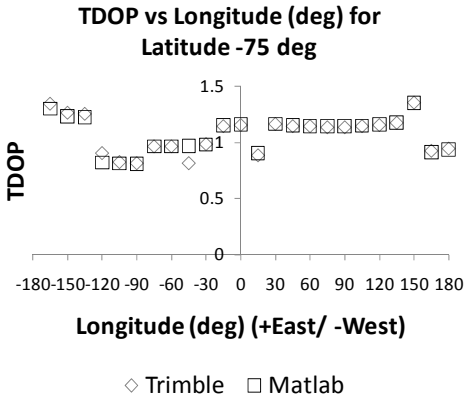

VDOP vs Longitude (deg) for Latitude 45 deg
VDOP
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

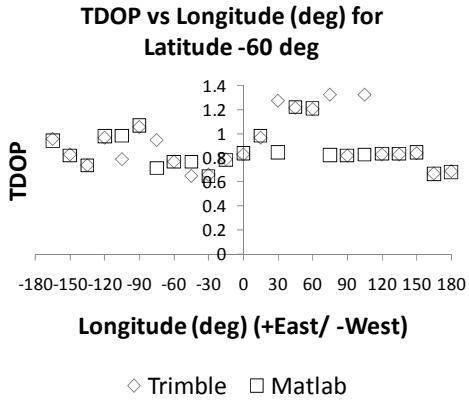

VDOP vs Longitude (deg) for Latitude 60 deg
VDOP
3.5
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

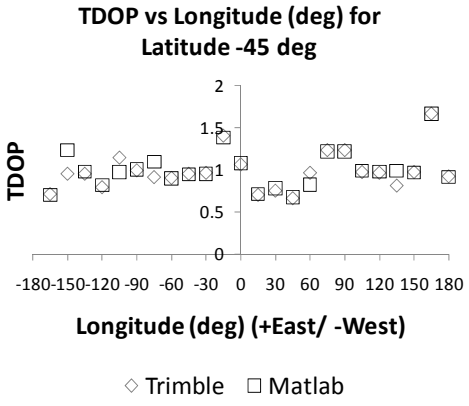

VDOP vs Longitude (deg) for Latitude 75 deg
VDOP
4
3
2
1
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

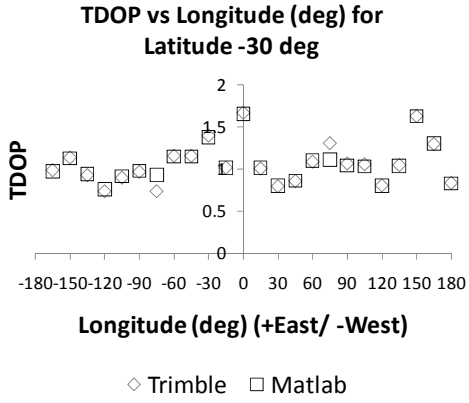
## B. TDOP

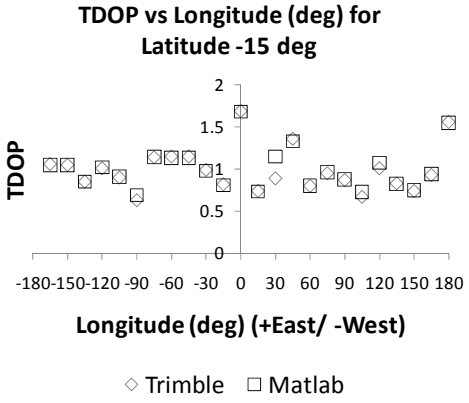

TDOP at North & South Pole
TDOP
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-90
0
90
Latitude (deg) (+North/ -South)
◇ Trimble □ Matlab

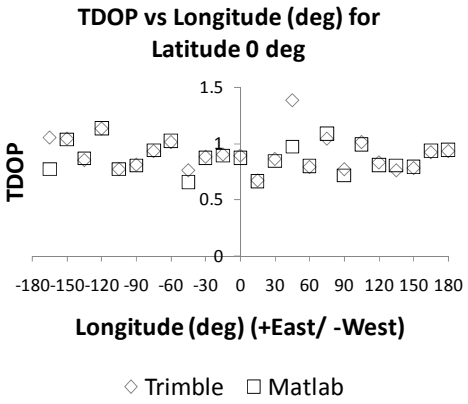

TDOP vs Longitude (deg) for Latitude -75 deg
TDOP
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

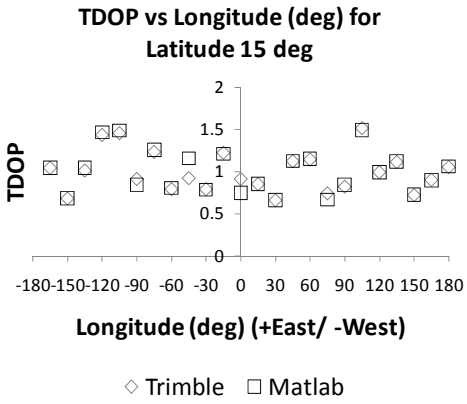

TDOP vs Longitude (deg) for Latitude -60 deg
TDOP
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

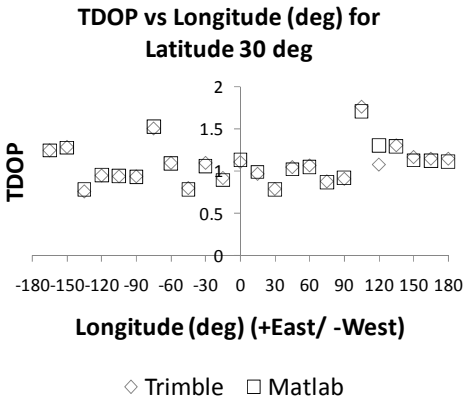

TDOP vs Longitude (deg) for Latitude -45 deg
TDOP
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

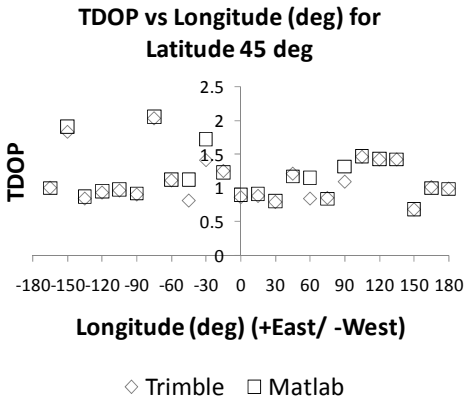

TDOP vs Longitude (deg) for Latitude -30 deg
TDOP
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

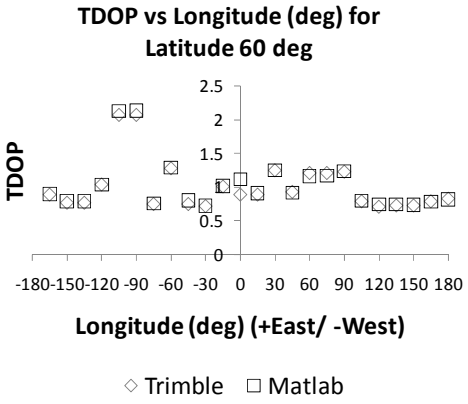

TDOP vs Longitude (deg) for Latitude -15 deg
TDOP
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

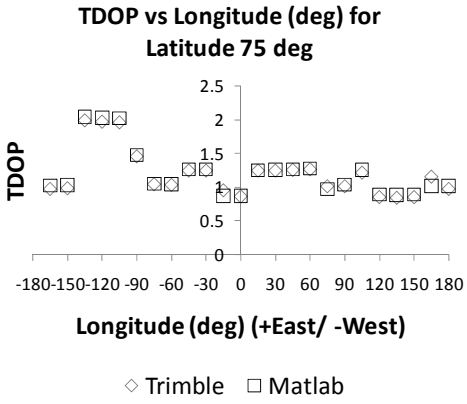

TDOP vs Longitude (deg) for Latitude 0 deg
TDOP
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab


TDOP vs Longitude (deg) for Latitude 15 deg
TDOP
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

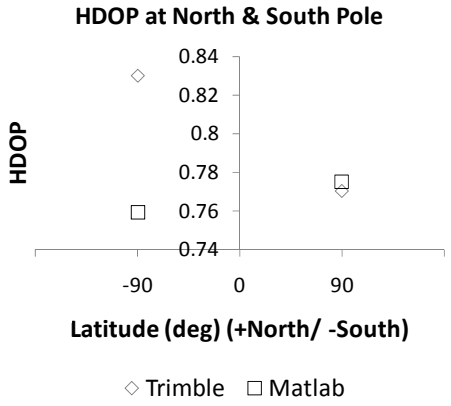

TDOP vs Longitude (deg) for Latitude 30 deg
TDOP
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

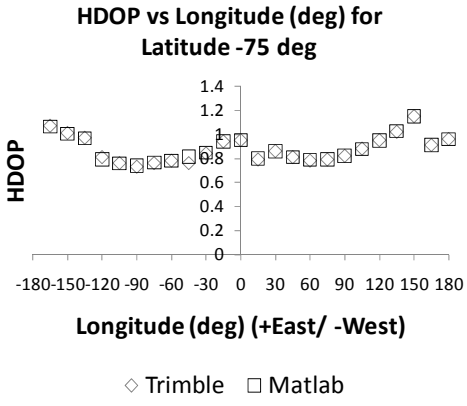

TDOP vs Longitude (deg) for Latitude 45 deg
TDOP
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

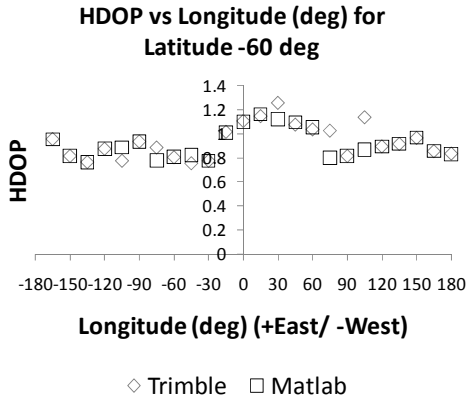

TDOP vs Longitude (deg) for Latitude 60 deg
TDOP
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

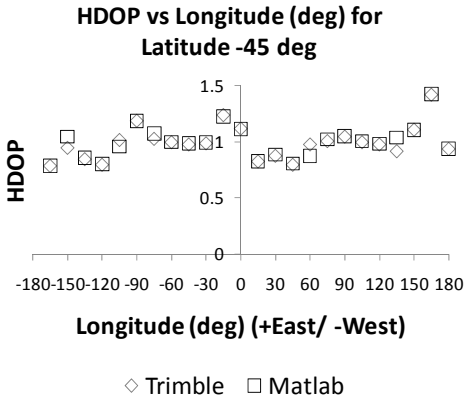

TDOP vs Longitude (deg) for Latitude 75 deg
TDOP
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

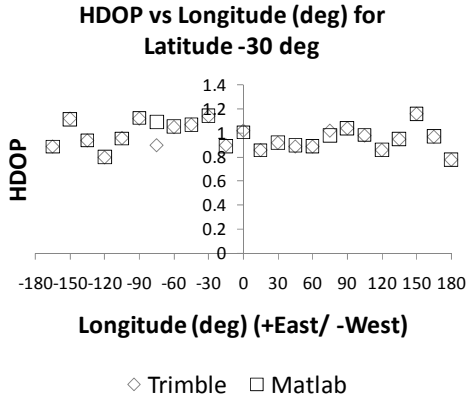
## C. HDOP

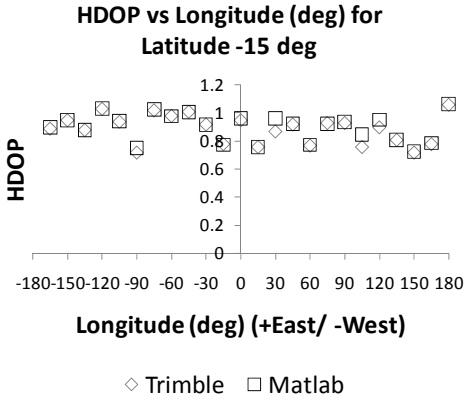

HDOP at North & South Pole
HDOP
0.84
0.82
0.8
0.78
0.76
0.74
-90
0
90
Latitude (deg) (+North/ -South)
◇ Trimble □ Matlab

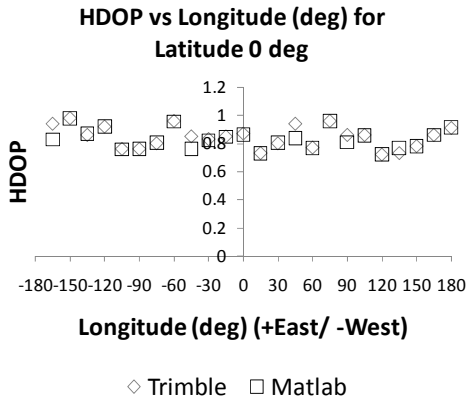

HDOP vs Longitude (deg) for Latitude -75 deg
HDOP
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

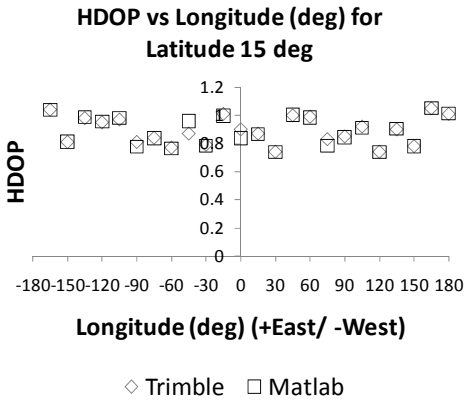

HDOP vs Longitude (deg) for Latitude -60 deg
HDOP
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

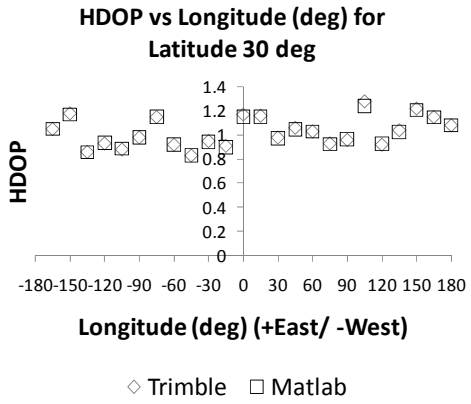

HDOP vs Longitude (deg) for Latitude -45 deg
HDOP
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

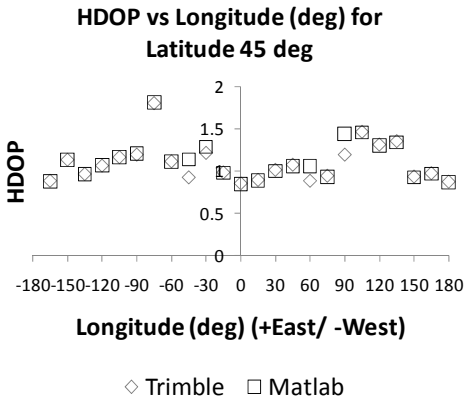

HDOP vs Longitude (deg) for Latitude -30 deg
HDOP
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

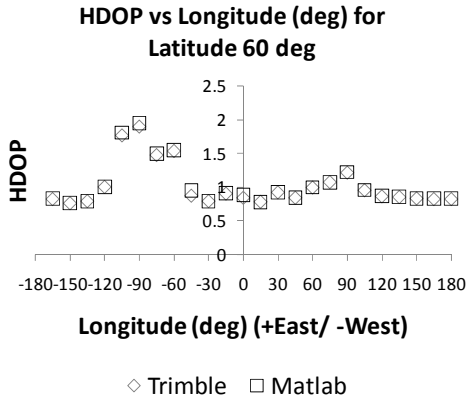

HDOP vs Longitude (deg) for Latitude -15 deg
HDOP
1.2
1
0.8
0.6
0.4
0.2
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

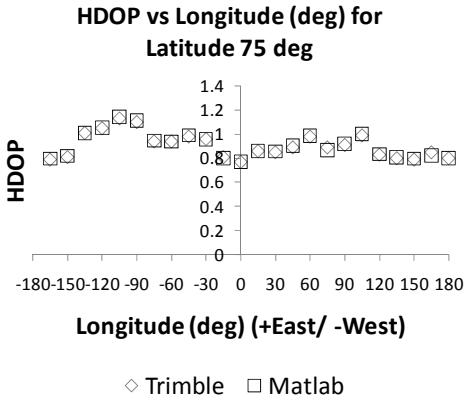

HDOP vs Longitude (deg) for Latitude 0 deg
HDOP
1.2
1
0.8
0.6
0.4
0.2
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab


HDOP vs Longitude (deg) for Latitude 15 deg
HDOP
1.2
1
0.8
0.6
0.4
0.2
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

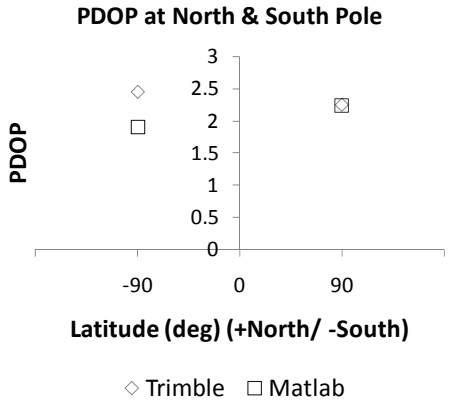

HDOP vs Longitude (deg) for Latitude 30 deg
HDOP
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

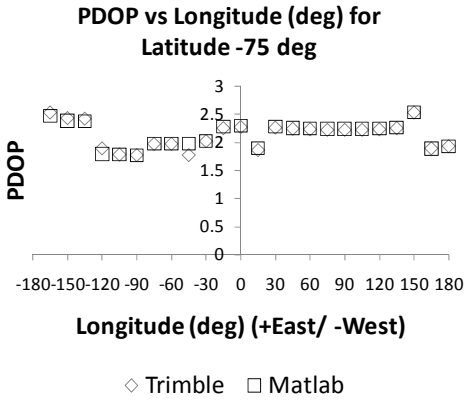

HDOP vs Longitude (deg) for Latitude 45 deg
HDOP
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

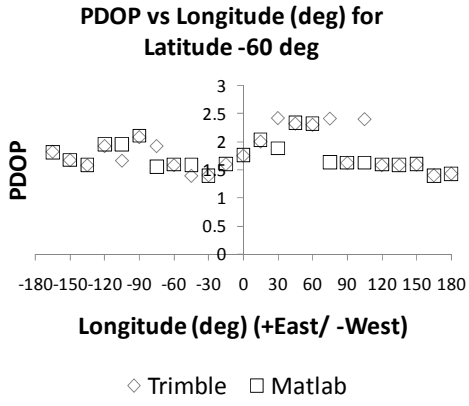

HDOP vs Longitude (deg) for Latitude 60 deg
HDOP
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

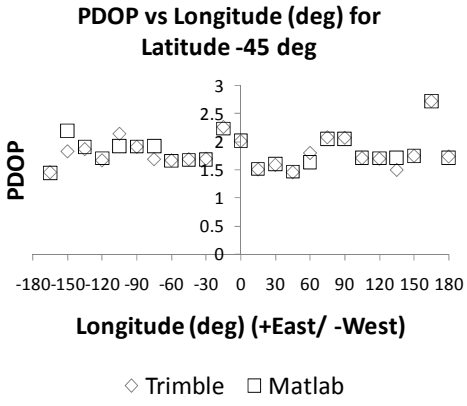

HDOP vs Longitude (deg) for Latitude 75 deg
HDOP
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

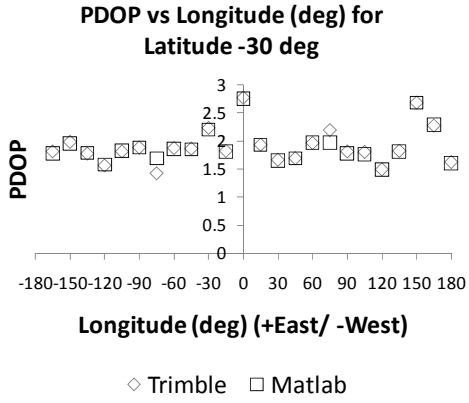
## D. PDOP

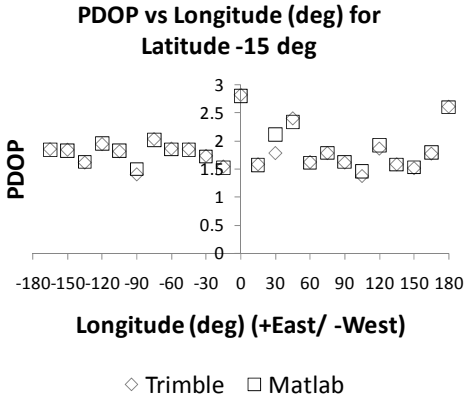

PDOP at North & South Pole
PDOP
3
2.5
2
1.5
1
0.5
0
-90
0
90
Latitude (deg) (+North/ -South)
◇ Trimble □ Matlab

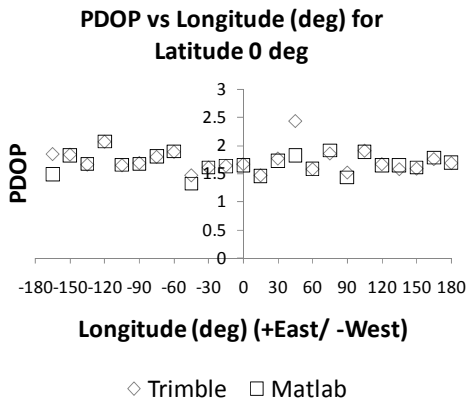

PDOP vs Longitude (deg) for Latitude -75 deg
PDOP
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

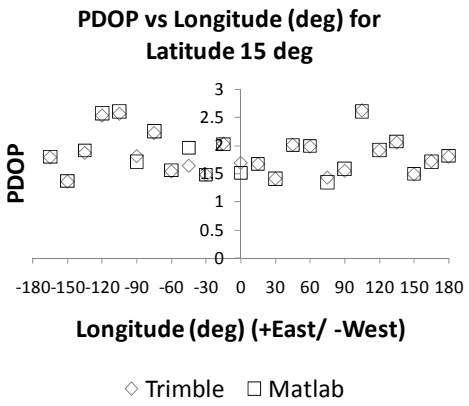

PDOP vs Longitude (deg) for Latitude -60 deg
PDOP
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

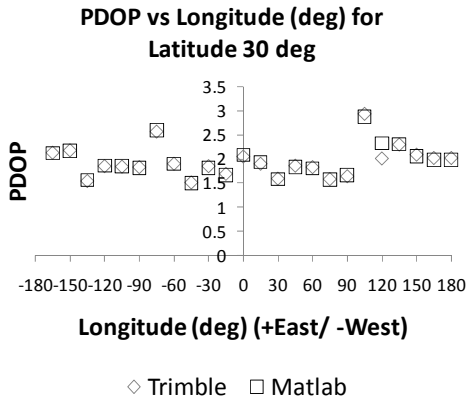

PDOP vs Longitude (deg) for Latitude -45 deg
PDOP
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

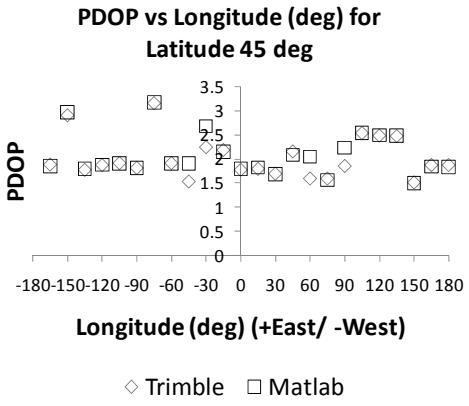

PDOP vs Longitude (deg) for Latitude -30 deg
PDOP
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

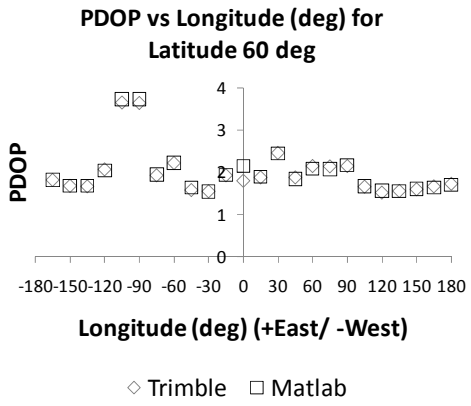

PDOP vs Longitude (deg) for Latitude -15 deg
PDOP
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

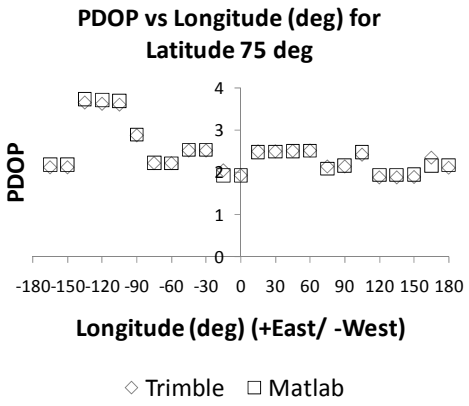

PDOP vs Longitude (deg) for Latitude 0 deg
PDOP
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab


PDOP vs Longitude (deg) for Latitude 15 deg
PDOP
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

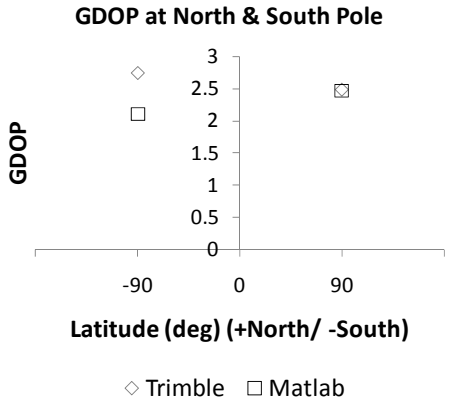

PDOP vs Longitude (deg) for Latitude 30 deg
PDOP
3.5
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

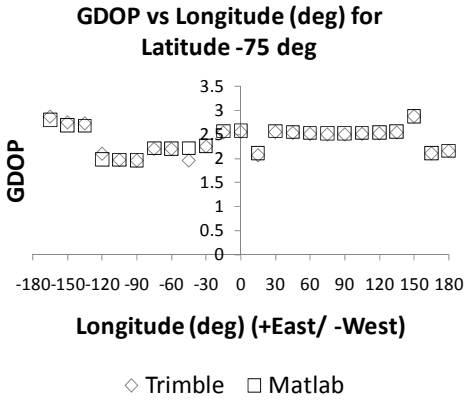

PDOP vs Longitude (deg) for Latitude 45 deg
PDOP
3.5
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

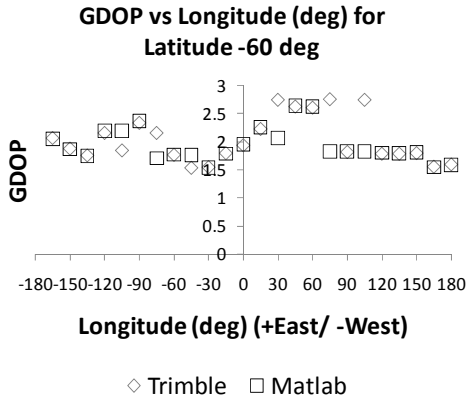

PDOP vs Longitude (deg) for Latitude 60 deg
PDOP
4
3
2
1
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

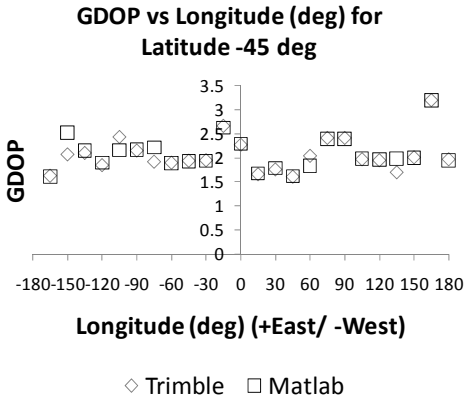

PDOP vs Longitude (deg) for Latitude 75 deg
PDOP
4
3
2
1
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

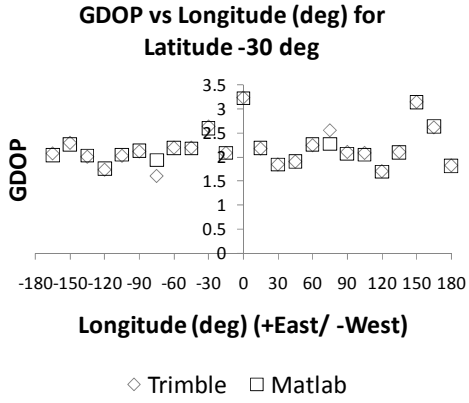
## E. GDOP

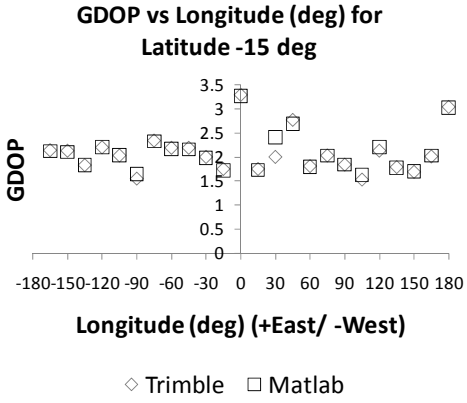

GDOP at North & South Pole
GDOP
3
2.5
2
1.5
1
0.5
0
-90
0
90
Latitude (deg) (+North/ -South)
◇ Trimble □ Matlab

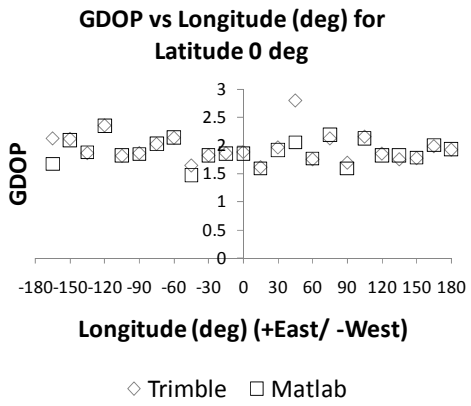

GDOP vs Longitude (deg) for Latitude -75 deg
GDOP
3.5
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

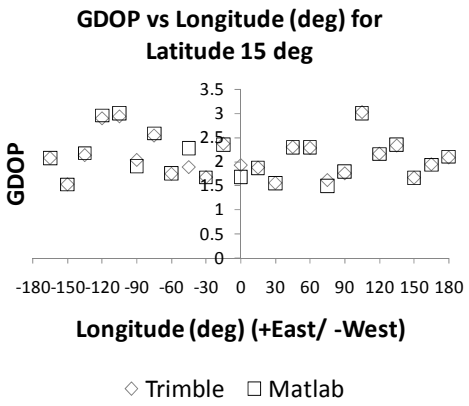

GDOP vs Longitude (deg) for Latitude -60 deg
GDOP
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

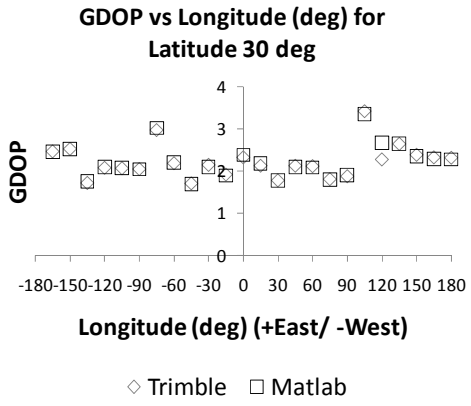

GDOP vs Longitude (deg) for Latitude -45 deg
GDOP
3.5
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

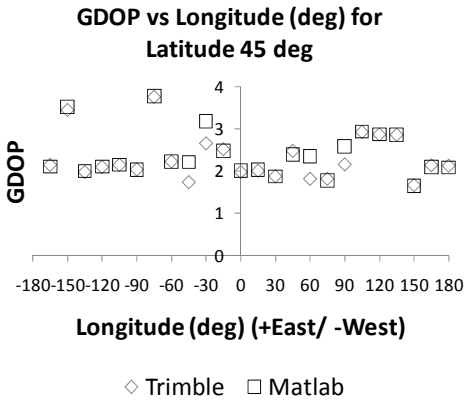

GDOP vs Longitude (deg) for Latitude -30 deg
GDOP
3.5
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

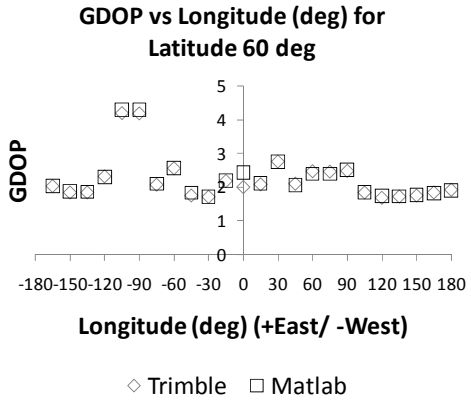

GDOP vs Longitude (deg) for Latitude -15 deg
GDOP
3.5
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

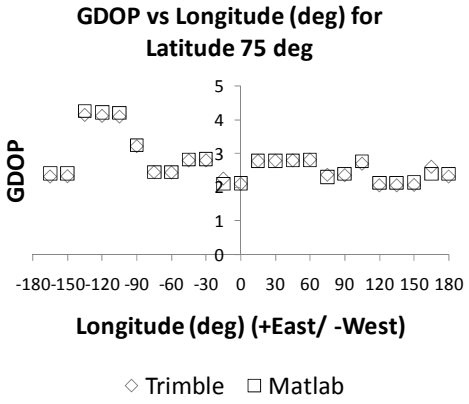

GDOP vs Longitude (deg) for Latitude 0 deg
GDOP
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab


GDOP vs Longitude (deg) for Latitude 15 deg
GDOP
3.5
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab


GDOP vs Longitude (deg) for Latitude 30 deg
GDOP
4
3
2
1
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab


GDOP vs Longitude (deg) for Latitude 45 deg
GDOP
4
3
2
1
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab


GDOP vs Longitude (deg) for Latitude 60 deg
GDOP
5
4
3
2
1
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab


GDOP vs Longitude (deg) for Latitude 75 deg
GDOP
5
4
3
2
1
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Matlab

## F. DOP COMPARISON IN TABLE

| Location | | Trimble | | | | | | Matlab | | | | | | Percentage Difference | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Lat (+/-90) +North/-South) | Long (+/-180) (+East/-West) | GDOP | PDOP | HDOP | TDOP | VDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | XDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | Difference in Visible SV |
| -75 | -165 | 2.86 | 2.53 | 1.07 | 1.34 | 2.29 | 8 | 2.790 | 2.470 | 1.063 | 1.296 | 2.230 | 0.882 | 0.592 | 8 | -2.47 | -2.36 | -0.70 | -3.30 | -2.62 | 0 |
| | -150 | 2.74 | 2.43 | 1.01 | 1.26 | 2.21 | 9 | 2.678 | 2.381 | 1.000 | 1.225 | 2.161 | 0.820 | 0.573 | 9 | -2.26 | -2.00 | -1.00 | -2.75 | -2.20 | 0 |
| | -135 | 2.72 | 2.42 | 0.97 | 1.25 | 2.22 | 9 | 2.668 | 2.374 | 0.964 | 1.219 | 2.169 | 0.787 | 0.557 | 9 | -1.90 | -1.91 | -0.63 | -2.52 | -2.28 | 0 |
| | -120 | 2.09 | 1.89 | 0.81 | 0.9 | 1.71 | 10 | 1.968 | 1.790 | 0.790 | 0.818 | 1.606 | 0.584 | 0.532 | 11 | -5.82 | -5.29 | -2.48 | -9.07 | -6.06 | 1 |
| | -105 | 1.96 | 1.78 | 0.76 | 0.82 | 1.61 | 11 | 1.955 | 1.779 | 0.755 | 0.810 | 1.611 | 0.536 | 0.532 | 11 | -0.28 | -0.07 | -0.64 | -1.20 | 0.03 | 0 |
| | -90 | 1.95 | 1.77 | 0.73 | 0.81 | 1.61 | 11 | 1.944 | 1.770 | 0.735 | 0.804 | 1.610 | 0.494 | 0.544 | 11 | -0.30 | 0.00 | 0.82 | -0.69 | 0.02 | 0 |
| | -75 | 2.19 | 1.97 | 0.76 | 0.96 | 1.82 | 10 | 2.200 | 1.979 | 0.763 | 0.962 | 1.826 | 0.513 | 0.565 | 10 | 0.47 | 0.46 | 0.33 | 0.20 | 0.34 | 0 |
| | -60 | 2.19 | 1.97 | 0.78 | 0.96 | 1.81 | 10 | 2.195 | 1.974 | 0.777 | 0.960 | 1.815 | 0.498 | 0.596 | 10 | 0.25 | 0.21 | -0.45 | 0.04 | 0.28 | 0 |
| | -45 | 1.95 | 1.77 | 0.76 | 0.81 | 1.6 | 11 | 2.200 | 1.977 | 0.812 | 0.964 | 1.802 | 0.499 | 0.641 | 10 | 12.80 | 11.69 | 6.86 | 19.06 | 12.65 | -1 |
| | -30 | 2.24 | 2.02 | 0.84 | 0.98 | 1.83 | 10 | 2.248 | 2.025 | 0.843 | 0.976 | 1.841 | 0.572 | 0.620 | 10 | 0.36 | 0.25 | 0.38 | -0.40 | 0.61 | 0 |
| | -15 | 2.53 | 2.25 | 0.94 | 1.14 | 2.05 | 9 | 2.555 | 2.282 | 0.937 | 1.148 | 2.081 | 0.626 | 0.697 | 9 | 0.98 | 1.44 | -0.30 | 0.68 | 1.51 | 0 |
| | 0 | 2.54 | 2.27 | 0.95 | 1.15 | 2.06 | 9 | 2.569 | 2.295 | 0.948 | 1.156 | 2.090 | 0.687 | 0.654 | 9 | 1.15 | 1.08 | -0.20 | 0.48 | 1.44 | 0 |
| | 15 | 2.06 | 1.86 | 0.79 | 0.88 | 1.68 | 10 | 2.094 | 1.891 | 0.794 | 0.900 | 1.716 | 0.613 | 0.505 | 10 | 1.66 | 1.66 | 0.56 | 2.28 | 2.14 | 0 |
| | 30 | 2.54 | 2.26 | 0.86 | 1.16 | 2.09 | 10 | 2.556 | 2.277 | 0.858 | 1.160 | 2.109 | 0.634 | 0.578 | 10 | 0.61 | 0.76 | -0.28 | 0.01 | 0.93 | 0 |
| | 45 | 2.52 | 2.24 | 0.81 | 1.14 | 2.09 | 10 | 2.535 | 2.260 | 0.808 | 1.148 | 2.110 | 0.617 | 0.522 | 10 | 0.60 | 0.89 | -0.21 | 0.74 | 0.98 | 0 |
| | 60 | 2.5 | 2.23 | 0.78 | 1.14 | 2.09 | 10 | 2.520 | 2.247 | 0.784 | 1.141 | 2.106 | 0.598 | 0.507 | 10 | 0.80 | 0.77 | 0.49 | 0.05 | 0.77 | 0 |
| | 75 | 2.49 | 2.22 | 0.79 | 1.13 | 2.08 | 10 | 2.512 | 2.239 | 0.788 | 1.137 | 2.096 | 0.575 | 0.538 | 10 | 0.87 | 0.87 | -0.25 | 0.65 | 0.78 | 0 |
| | 90 | 2.49 | 2.22 | 0.82 | 1.13 | 2.06 | 10 | 2.510 | 2.237 | 0.820 | 1.139 | 2.082 | 0.556 | 0.603 | 10 | 0.82 | 0.78 | 0.01 | 0.76 | 1.05 | 0 |
| | 105 | 2.5 | 2.22 | 0.88 | 1.14 | 2.04 | 10 | 2.516 | 2.241 | 0.875 | 1.145 | 2.063 | 0.553 | 0.678 | 10 | 0.66 | 0.95 | -0.57 | 0.40 | 1.14 | 0 |
| | 120 | 2.51 | 2.23 | 0.95 | 1.15 | 2.02 | 10 | 2.530 | 2.251 | 0.945 | 1.155 | 2.043 | 0.584 | 0.743 | 10 | 0.78 | 0.92 | -0.51 | 0.43 | 1.11 | 0 |
| | 135 | 2.53 | 2.25 | 1.02 | 1.17 | 2 | 10 | 2.550 | 2.266 | 1.023 | 1.169 | 2.021 | 0.656 | 0.786 | 10 | 0.77 | 0.69 | 0.31 | -0.08 | 1.07 | 0 |
| | 150 | 2.87 | 2.53 | 1.15 | 1.35 | 2.25 | 9 | 2.870 | 2.534 | 1.149 | 1.348 | 2.258 | 0.819 | 0.805 | 9 | 0.00 | 0.14 | -0.11 | -0.15 | 0.37 | 0 |
| | 165 | 2.1 | 1.89 | 0.91 | 0.92 | 1.66 | 10 | 2.095 | 1.886 | 0.907 | 0.911 | 1.654 | 0.728 | 0.542 | 10 | -0.25 | -0.20 | -0.32 | -1.00 | -0.37 | 0 |
| | 180 | 2.15 | 1.93 | 0.96 | 0.94 | 1.68 | 9 | 2.144 | 1.930 | 0.958 | 0.934 | 1.675 | 0.807 | 0.516 | 9 | -0.29 | -0.02 | -0.25 | -0.66 | -0.28 | 0 |

| Location | | Trimble | | | | | | Matlab | | | | | | | | Percentage Difference | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Lat (+/-90) (+North/-South) | Long (+/-180) (+East/-West) | GDOP | PDOP | HDOP | TDOP | VDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | Difference in Visible SV |
| -60 | -165 | 2.07 | 1.83 | 0.96 | 0.96 | 1.56 | 8 | 2.054 | 1.822 | 0.959 | 0.947 | 1.550 | 0.690 | 0.666 | 8 | -0.80 | -0.44 | -0.16 | -1.33 | -0.67 | 0 |
| | -150 | 1.88 | 1.68 | 0.82 | 0.83 | 1.47 | 9 | 1.874 | 1.684 | 0.818 | 0.822 | 1.472 | 0.589 | 0.568 | 9 | -0.34 | 0.22 | -0.22 | -0.98 | 0.10 | 0 |
| | -135 | 1.76 | 1.59 | 0.77 | 0.74 | 1.4 | 10 | 1.758 | 1.595 | 0.765 | 0.741 | 1.399 | 0.587 | 0.491 | 10 | -0.10 | 0.29 | -0.62 | 0.12 | -0.07 | 0 |
| | -120 | 2.16 | 1.93 | 0.88 | 0.97 | 1.72 | 9 | 2.195 | 1.961 | 0.880 | 0.985 | 1.753 | 0.715 | 0.514 | 9 | 1.61 | 1.62 | 0.00 | 1.53 | 1.91 | 0 |
| | -105 | 1.85 | 1.67 | 0.78 | 0.79 | 1.48 | 10 | 2.200 | 1.966 | 0.890 | 0.988 | 1.753 | 0.694 | 0.556 | 9 | 18.91 | 17.69 | 14.06 | 25.04 | 18.43 | -1 |
| | -90 | 2.34 | 2.09 | 0.94 | 1.06 | 1.87 | 9 | 2.368 | 2.111 | 0.938 | 1.073 | 1.892 | 0.727 | 0.592 | 9 | 1.21 | 1.01 | -0.26 | 1.26 | 1.16 | 0 |
| | -75 | 2.16 | 1.93 | 0.89 | 0.95 | 1.72 | 10 | 1.717 | 1.560 | 0.784 | 0.718 | 1.348 | 0.556 | 0.552 | 11 | -20.51 | -19.20 | -11.93 | -24.40 | -21.62 | 1 |
| | -60 | 1.77 | 1.59 | 0.81 | 0.77 | 1.37 | 10 | 1.780 | 1.604 | 0.811 | 0.772 | 1.384 | 0.557 | 0.589 | 10 | 0.54 | 0.86 | 0.11 | 0.21 | 0.99 | 0 |
| | -45 | 1.54 | 1.4 | 0.76 | 0.65 | 1.18 | 11 | 1.775 | 1.599 | 0.826 | 0.771 | 1.369 | 0.568 | 0.600 | 10 | 15.27 | 14.22 | 8.64 | 18.55 | 16.05 | -1 |
| | -30 | 1.55 | 1.41 | 0.78 | 0.66 | 1.17 | 11 | 1.548 | 1.404 | 0.779 | 0.653 | 1.168 | 0.504 | 0.594 | 11 | -0.11 | -0.43 | -0.15 | -1.11 | -0.15 | 0 |
| | -15 | 1.8 | 1.61 | 1.02 | 0.79 | 1.25 | 9 | 1.795 | 1.613 | 1.013 | 0.787 | 1.255 | 0.715 | 0.717 | 9 | -0.29 | 0.18 | -0.72 | -0.38 | 0.42 | 0 |
| | 0 | 1.94 | 1.76 | 1.11 | 0.83 | 1.36 | 8 | 1.965 | 1.775 | 1.106 | 0.843 | 1.388 | 0.921 | 0.611 | 8 | 1.28 | 0.85 | -0.40 | 1.55 | 2.09 | 0 |
| | 15 | 2.23 | 2.01 | 1.15 | 0.97 | 1.64 | 8 | 2.268 | 2.042 | 1.166 | 0.988 | 1.676 | 1.044 | 0.520 | 8 | 1.71 | 1.58 | 1.38 | 1.82 | 2.21 | 0 |
| | 30 | 2.75 | 2.43 | 1.26 | 1.28 | 2.08 | 8 | 2.070 | 1.888 | 1.125 | 0.849 | 1.516 | 1.033 | 0.446 | 9 | -24.74 | -22.32 | -10.70 | -33.71 | -27.13 | 1 |
| | 45 | 2.63 | 2.33 | 1.08 | 1.22 | 2.06 | 9 | 2.645 | 2.343 | 1.099 | 1.226 | 2.070 | 0.963 | 0.530 | 9 | 0.56 | 0.57 | 1.77 | 0.48 | 0.47 | 0 |
| | 60 | 2.61 | 2.31 | 1.04 | 1.21 | 2.06 | 9 | 2.624 | 2.326 | 1.059 | 1.216 | 2.071 | 0.916 | 0.532 | 9 | 0.54 | 0.68 | 1.82 | 0.48 | 0.51 | 0 |
| | 75 | 2.76 | 2.42 | 1.03 | 1.33 | 2.2 | 9 | 1.839 | 1.642 | 0.805 | 0.828 | 1.431 | 0.564 | 0.574 | 10 | -33.37 | -32.14 | -21.84 | -37.77 | -34.94 | 1 |
| | 90 | 1.82 | 1.63 | 0.82 | 0.82 | 1.41 | 10 | 1.830 | 1.634 | 0.818 | 0.824 | 1.415 | 0.561 | 0.596 | 10 | 0.57 | 0.26 | -0.21 | 0.50 | 0.33 | 0 |
| | 105 | 2.75 | 2.41 | 1.14 | 1.33 | 2.12 | 9 | 1.836 | 1.637 | 0.873 | 0.830 | 1.385 | 0.580 | 0.652 | 10 | -33.25 | -32.07 | -23.46 | -37.59 | -34.66 | 1 |
| | 120 | 1.79 | 1.59 | 0.9 | 0.83 | 1.31 | 9 | 1.809 | 1.604 | 0.897 | 0.837 | 1.330 | 0.639 | 0.629 | 9 | 1.08 | 0.89 | -0.38 | 0.87 | 1.53 | 0 |
| | 135 | 1.79 | 1.59 | 0.92 | 0.83 | 1.29 | 9 | 1.808 | 1.601 | 0.921 | 0.839 | 1.310 | 0.649 | 0.653 | 9 | 0.99 | 0.72 | 0.10 | 1.02 | 1.56 | 0 |
| | 150 | 1.8 | 1.6 | 0.97 | 0.84 | 1.27 | 9 | 1.821 | 1.611 | 0.972 | 0.849 | 1.284 | 0.674 | 0.700 | 9 | 1.14 | 0.66 | 0.19 | 1.08 | 1.13 | 0 |
| | 165 | 1.56 | 1.41 | 0.86 | 0.67 | 1.11 | 9 | 1.560 | 1.409 | 0.862 | 0.670 | 1.115 | 0.622 | 0.597 | 9 | 0.01 | -0.06 | 0.24 | -0.04 | 0.41 | 0 |
| | 180 | 1.6 | 1.44 | 0.84 | 0.69 | 1.17 | 10 | 1.594 | 1.440 | 0.834 | 0.684 | 1.173 | 0.663 | 0.506 | 10 | -0.38 | -0.03 | -0.70 | -0.83 | 0.27 | 0 |

| Location | | Trimble | | | | | | Matlab | | | | | | | | Percentage Difference | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Lat (+/-90) (+North/-South) | Long (+/-180) (+East/-West) | GDOP | PDOP | HDOP | TDOP | VDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | Difference in Visible SV |
| -45 | -165 | 1.64 | 1.47 | 0.79 | 0.72 | 1.25 | 10 | 1.618 | 1.456 | 0.785 | 0.706 | 1.226 | 0.558 | 0.552 | 10 | -1.35 | -0.99 | -0.70 | -1.90 | -1.92 | 0 |
| | -150 | 2.08 | 1.84 | 0.95 | 0.96 | 1.57 | 8 | 2.533 | 2.209 | 1.044 | 1.240 | 1.947 | 0.678 | 0.795 | 7 | 21.79 | 20.05 | 9.94 | 29.17 | 23.98 | -1 |
| | -135 | 2.11 | 1.88 | 0.85 | 0.96 | 1.68 | 9 | 2.156 | 1.921 | 0.857 | 0.980 | 1.719 | 0.644 | 0.565 | 9 | 2.19 | 2.16 | 0.79 | 2.08 | 2.32 | 0 |
| | -120 | 1.86 | 1.68 | 0.8 | 0.8 | 1.47 | 10 | 1.908 | 1.722 | 0.802 | 0.824 | 1.523 | 0.652 | 0.468 | 10 | 2.60 | 2.47 | 0.30 | 2.96 | 3.61 | 0 |
| | -105 | 2.44 | 2.15 | 1.02 | 1.15 | 1.89 | 8 | 2.169 | 1.937 | 0.959 | 0.976 | 1.683 | 0.793 | 0.538 | 9 | -11.10 | -9.90 | -6.02 | -15.13 | -10.94 | 1 |
| | -90 | 2.16 | 1.92 | 1.19 | 1 | 1.5 | 8 | 2.174 | 1.926 | 1.186 | 1.009 | 1.517 | 1.044 | 0.563 | 8 | 0.66 | 0.32 | -0.31 | 0.87 | 1.16 | 0 |
| | -75 | 1.93 | 1.7 | 1.03 | 0.92 | 1.35 | 9 | 2.228 | 1.938 | 1.072 | 1.099 | 1.615 | 0.856 | 0.645 | 8 | 15.44 | 14.01 | 4.10 | 19.43 | 19.60 | -1 |
| | -60 | 1.91 | 1.68 | 1 | 0.91 | 1.35 | 9 | 1.899 | 1.671 | 0.999 | 0.903 | 1.339 | 0.763 | 0.645 | 9 | -0.58 | -0.56 | -0.09 | -0.80 | -0.81 | 0 |
| | -45 | 1.95 | 1.7 | 0.98 | 0.96 | 1.39 | 9 | 1.938 | 1.690 | 0.987 | 0.950 | 1.371 | 0.753 | 0.638 | 9 | -0.60 | -0.61 | 0.73 | -1.08 | -1.35 | 0 |
| | -30 | 1.96 | 1.71 | 0.99 | 0.97 | 1.39 | 9 | 1.949 | 1.699 | 0.993 | 0.955 | 1.379 | 0.769 | 0.628 | 9 | -0.55 | -0.62 | 0.32 | -1.56 | -0.80 | 0 |
| | -15 | 2.67 | 2.27 | 1.24 | 1.41 | 1.9 | 8 | 2.637 | 2.243 | 1.228 | 1.387 | 1.877 | 0.984 | 0.734 | 8 | -1.24 | -1.20 | -1.00 | -1.64 | -1.21 | 0 |
| | 0 | 2.28 | 2.02 | 1.12 | 1.07 | 1.67 | 9 | 2.300 | 2.029 | 1.113 | 1.085 | 1.696 | 0.962 | 0.561 | 9 | 0.89 | 0.42 | -0.60 | 1.38 | 1.53 | 0 |
| | 15 | 1.67 | 1.52 | 0.83 | 0.71 | 1.27 | 10 | 1.688 | 1.529 | 0.826 | 0.714 | 1.287 | 0.660 | 0.496 | 10 | 1.05 | 0.60 | -0.52 | 0.54 | 1.34 | 0 |
| | 30 | 1.77 | 1.6 | 0.88 | 0.76 | 1.33 | 9 | 1.800 | 1.621 | 0.886 | 0.782 | 1.357 | 0.710 | 0.531 | 9 | 1.67 | 1.30 | 0.70 | 2.91 | 2.03 | 0 |
| | 45 | 1.62 | 1.47 | 0.8 | 0.67 | 1.23 | 10 | 1.627 | 1.481 | 0.806 | 0.676 | 1.242 | 0.640 | 0.489 | 10 | 0.45 | 0.71 | 0.72 | 0.82 | 0.97 | 0 |
| | 60 | 2.05 | 1.81 | 0.98 | 0.97 | 1.51 | 9 | 1.843 | 1.647 | 0.876 | 0.826 | 1.395 | 0.664 | 0.572 | 9 | -10.11 | -8.99 | -10.64 | -14.81 | -7.60 | 0 |
| | 75 | 2.42 | 2.09 | 1.01 | 1.24 | 1.82 | 8 | 2.400 | 2.066 | 1.022 | 1.221 | 1.796 | 0.786 | 0.653 | 8 | -0.82 | -1.14 | 1.15 | -1.52 | -1.32 | 0 |
| | 90 | 2.42 | 2.08 | 1.05 | 1.24 | 1.8 | 8 | 2.397 | 2.061 | 1.052 | 1.223 | 1.773 | 0.782 | 0.703 | 8 | -0.95 | -0.89 | 0.20 | -1.38 | -1.52 | 0 |
| | 105 | 1.98 | 1.72 | 1 | 0.98 | 1.4 | 9 | 1.994 | 1.730 | 1.004 | 0.992 | 1.409 | 0.735 | 0.685 | 9 | 0.73 | 0.60 | 0.43 | 1.18 | 0.66 | 0 |
| | 120 | 1.96 | 1.71 | 0.98 | 0.97 | 1.4 | 9 | 1.980 | 1.720 | 0.982 | 0.981 | 1.412 | 0.765 | 0.615 | 9 | 1.02 | 0.58 | 0.20 | 1.12 | 0.86 | 0 |
| | 135 | 1.71 | 1.51 | 0.92 | 0.82 | 1.19 | 10 | 1.997 | 1.733 | 1.039 | 0.993 | 1.386 | 0.788 | 0.678 | 9 | 16.77 | 14.74 | 12.95 | 21.06 | 16.50 | -1 |
| | 150 | 2.02 | 1.76 | 1.11 | 0.98 | 1.37 | 9 | 2.011 | 1.761 | 1.107 | 0.971 | 1.369 | 0.941 | 0.583 | 9 | -0.46 | 0.03 | -0.26 | -0.91 | -0.07 | 0 |
| | 165 | 3.2 | 2.73 | 1.42 | 1.67 | 2.33 | 9 | 3.206 | 2.734 | 1.427 | 1.674 | 2.332 | 1.289 | 0.614 | 9 | 0.17 | 0.14 | 0.52 | 0.23 | 0.06 | 0 |
| | 180 | 1.98 | 1.75 | 0.94 | 0.93 | 1.47 | 10 | 1.955 | 1.726 | 0.938 | 0.918 | 1.449 | 0.773 | 0.531 | 10 | -1.29 | -1.38 | -0.24 | -1.33 | -1.44 | 0 |

| Location | | Trimble | | | | | | Matlab | | | | | | | | Percentage Difference | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Lat (+/-90) (+North/-South) | Long (+/-180) (+East/-West) | GDOP | PDOP | HDOP | TDOP | VDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | Difference in Visible SV |
| -30 | -165 | 2.08 | 1.82 | 0.89 | 0.99 | 1.59 | 10 | 2.036 | 1.790 | 0.889 | 0.970 | 1.554 | 0.569 | 0.684 | 10 | -2.10 | -1.63 | -0.10 | -2.00 | -2.27 | 0 |
| | -150 | 2.3 | 1.99 | 1.12 | 1.14 | 1.64 | 9 | 2.263 | 1.963 | 1.113 | 1.127 | 1.617 | 0.611 | 0.930 | 9 | -1.59 | -1.36 | -0.60 | -1.15 | -1.42 | 0 |
| | -135 | 2.01 | 1.78 | 0.94 | 0.93 | 1.52 | 8 | 2.028 | 1.797 | 0.940 | 0.939 | 1.532 | 0.626 | 0.701 | 8 | 0.87 | 0.96 | 0.00 | 0.95 | 0.76 | 0 |
| | -120 | 1.74 | 1.57 | 0.8 | 0.74 | 1.35 | 10 | 1.763 | 1.592 | 0.801 | 0.756 | 1.376 | 0.616 | 0.513 | 10 | 1.30 | 1.42 | 0.15 | 2.16 | 1.93 | 0 |
| | -105 | 2.03 | 1.82 | 0.96 | 0.9 | 1.55 | 9 | 2.054 | 1.838 | 0.955 | 0.916 | 1.571 | 0.776 | 0.556 | 9 | 1.17 | 1.01 | -0.57 | 1.76 | 1.36 | 0 |
| | -90 | 2.12 | 1.89 | 1.12 | 0.97 | 1.51 | 8 | 2.130 | 1.893 | 1.126 | 0.977 | 1.522 | 0.978 | 0.558 | 8 | 0.48 | 0.15 | 0.54 | 0.73 | 0.77 | 0 |
| | -75 | 1.61 | 1.43 | 0.9 | 0.74 | 1.12 | 10 | 1.942 | 1.704 | 1.094 | 0.931 | 1.306 | 0.883 | 0.645 | 9 | 20.59 | 19.13 | 21.53 | 25.86 | 16.60 | -1 |
| | -60 | 2.2 | 1.88 | 1.05 | 1.16 | 1.55 | 9 | 2.191 | 1.865 | 1.056 | 1.150 | 1.537 | 0.781 | 0.710 | 9 | -0.42 | -0.81 | 0.53 | -0.87 | -0.83 | 0 |
| | -45 | 2.2 | 1.87 | 1.07 | 1.16 | 1.54 | 9 | 2.184 | 1.858 | 1.068 | 1.148 | 1.521 | 0.766 | 0.744 | 9 | -0.71 | -0.62 | -0.18 | -1.05 | -1.25 | 0 |
| | -30 | 2.64 | 2.24 | 1.14 | 1.4 | 1.92 | 8 | 2.607 | 2.214 | 1.144 | 1.377 | 1.896 | 0.825 | 0.792 | 8 | -1.25 | -1.17 | 0.32 | -1.65 | -1.28 | 0 |
| | -15 | 2.09 | 1.83 | 0.9 | 1.02 | 1.59 | 10 | 2.088 | 1.823 | 0.894 | 1.018 | 1.589 | 0.687 | 0.573 | 10 | -0.10 | -0.37 | -0.64 | -0.23 | -0.07 | 0 |
| | 0 | 3.24 | 2.78 | 1.02 | 1.67 | 2.58 | 9 | 3.227 | 2.767 | 1.008 | 1.659 | 2.578 | 0.765 | 0.656 | 9 | -0.41 | -0.45 | -1.22 | -0.63 | -0.10 | 0 |
| | 15 | 2.18 | 1.94 | 0.86 | 1.01 | 1.74 | 10 | 2.192 | 1.943 | 0.859 | 1.015 | 1.742 | 0.641 | 0.572 | 10 | 0.54 | 0.13 | -0.09 | 0.51 | 0.13 | 0 |
| | 30 | 1.85 | 1.67 | 0.92 | 0.8 | 1.4 | 9 | 1.844 | 1.661 | 0.922 | 0.802 | 1.381 | 0.641 | 0.662 | 9 | -0.33 | -0.57 | 0.18 | 0.21 | -1.34 | 0 |
| | 45 | 1.91 | 1.7 | 0.89 | 0.86 | 1.45 | 10 | 1.906 | 1.700 | 0.900 | 0.862 | 1.443 | 0.645 | 0.628 | 10 | -0.19 | 0.02 | 1.15 | 0.23 | -0.51 | 0 |
| | 60 | 2.25 | 1.97 | 0.89 | 1.09 | 1.76 | 9 | 2.265 | 1.979 | 0.892 | 1.102 | 1.767 | 0.658 | 0.603 | 9 | 0.68 | 0.48 | 0.24 | 1.10 | 0.40 | 0 |
| | 75 | 2.56 | 2.2 | 1.02 | 1.31 | 1.95 | 8 | 2.275 | 1.984 | 0.981 | 1.114 | 1.725 | 0.657 | 0.729 | 9 | -11.12 | -9.81 | -3.80 | -14.98 | -11.55 | 1 |
| | 90 | 2.11 | 1.82 | 1.04 | 1.07 | 1.49 | 9 | 2.068 | 1.786 | 1.038 | 1.042 | 1.454 | 0.705 | 0.761 | 9 | -1.99 | -1.85 | -0.23 | -2.64 | -2.41 | 0 |
| | 105 | 2.09 | 1.81 | 0.98 | 1.06 | 1.52 | 9 | 2.053 | 1.776 | 0.985 | 1.029 | 1.478 | 0.735 | 0.656 | 9 | -1.79 | -1.88 | 0.54 | -2.93 | -2.79 | 0 |
| | 120 | 1.71 | 1.5 | 0.86 | 0.81 | 1.23 | 10 | 1.698 | 1.498 | 0.862 | 0.800 | 1.225 | 0.658 | 0.557 | 10 | -0.70 | -0.16 | 0.20 | -1.21 | -0.42 | 0 |
| | 135 | 2.11 | 1.83 | 0.95 | 1.05 | 1.56 | 9 | 2.094 | 1.819 | 0.949 | 1.036 | 1.552 | 0.741 | 0.593 | 9 | -0.78 | -0.58 | -0.09 | -1.35 | -0.50 | 0 |
| | 150 | 3.15 | 2.69 | 1.16 | 1.63 | 2.43 | 9 | 3.145 | 2.691 | 1.160 | 1.628 | 2.428 | 0.773 | 0.865 | 9 | -0.17 | 0.02 | -0.03 | -0.12 | -0.09 | 0 |
| | 165 | 2.62 | 2.28 | 0.97 | 1.3 | 2.06 | 10 | 2.642 | 2.296 | 0.974 | 1.306 | 2.080 | 0.762 | 0.606 | 10 | 0.84 | 0.72 | 0.39 | 0.47 | 0.96 | 0 |
| | 180 | 1.83 | 1.63 | 0.78 | 0.84 | 1.43 | 11 | 1.813 | 1.614 | 0.779 | 0.826 | 1.413 | 0.572 | 0.529 | 11 | -0.95 | -1.00 | -0.08 | -1.70 | -1.19 | 0 |

| Location | | Trimble | | | | | | Matlab | | | | | | | | Percentage Difference | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Lat (+/-90) (+North/-South) | Long (+/-180) (+East/-West) | GDOP | PDOP | HDOP | TDOP | VDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | Difference in Visible SV |
| -15 | -165 | 2.14 | 1.86 | 0.89 | 1.06 | 1.63 | 9 | 2.127 | 1.849 | 0.897 | 1.051 | 1.617 | 0.628 | 0.640 | 9 | -0.62 | -0.61 | 0.78 | -0.81 | -0.82 | 0 |
| | -150 | 2.13 | 1.85 | 0.95 | 1.05 | 1.59 | 9 | 2.114 | 1.837 | 0.947 | 1.046 | 1.574 | 0.631 | 0.706 | 9 | -0.77 | -0.72 | -0.37 | -0.36 | -1.01 | 0 |
| | -135 | 1.83 | 1.62 | 0.88 | 0.85 | 1.37 | 9 | 1.841 | 1.632 | 0.876 | 0.853 | 1.377 | 0.630 | 0.608 | 9 | 0.62 | 0.72 | -0.49 | 0.35 | 0.50 | 0 |
| | -120 | 2.2 | 1.95 | 1.03 | 1.01 | 1.66 | 8 | 2.215 | 1.965 | 1.030 | 1.023 | 1.673 | 0.750 | 0.706 | 8 | 0.70 | 0.77 | 0.02 | 1.29 | 0.80 | 0 |
| | -105 | 2.03 | 1.82 | 0.94 | 0.9 | 1.56 | 9 | 2.049 | 1.835 | 0.942 | 0.911 | 1.575 | 0.757 | 0.561 | 9 | 0.92 | 0.82 | 0.22 | 1.24 | 0.94 | 0 |
| | -90 | 1.55 | 1.41 | 0.72 | 0.63 | 1.22 | 12 | 1.659 | 1.507 | 0.752 | 0.693 | 1.306 | 0.541 | 0.522 | 11 | 7.02 | 6.89 | 4.46 | 9.97 | 7.06 | -1 |
| | -75 | 2.34 | 2.04 | 1.02 | 1.14 | 1.76 | 9 | 2.332 | 2.031 | 1.024 | 1.146 | 1.755 | 0.789 | 0.652 | 9 | -0.34 | -0.43 | 0.37 | 0.50 | -0.31 | 0 |
| | -60 | 2.19 | 1.87 | 0.98 | 1.15 | 1.6 | 9 | 2.177 | 1.857 | 0.977 | 1.137 | 1.580 | 0.670 | 0.711 | 9 | -0.58 | -0.68 | -0.32 | -1.17 | -1.28 | 0 |
| | -45 | 2.19 | 1.86 | 1.01 | 1.15 | 1.57 | 9 | 2.170 | 1.850 | 1.005 | 1.135 | 1.553 | 0.665 | 0.753 | 9 | -0.91 | -0.56 | -0.49 | -1.32 | -1.10 | 0 |
| | -30 | 2 | 1.74 | 0.92 | 0.99 | 1.48 | 10 | 1.988 | 1.729 | 0.915 | 0.980 | 1.467 | 0.650 | 0.644 | 10 | -0.61 | -0.61 | -0.51 | -0.97 | -0.86 | 0 |
| | -15 | 1.75 | 1.55 | 0.78 | 0.82 | 1.34 | 11 | 1.730 | 1.531 | 0.774 | 0.806 | 1.321 | 0.558 | 0.537 | 11 | -1.14 | -1.23 | -0.79 | -1.72 | -1.42 | 0 |
| | 0 | 3.29 | 2.83 | 0.95 | 1.69 | 2.66 | 9 | 3.280 | 2.815 | 0.958 | 1.683 | 2.647 | 0.698 | 0.656 | 9 | -0.30 | -0.52 | 0.88 | -0.40 | -0.48 | 0 |
| | 15 | 1.75 | 1.59 | 0.76 | 0.74 | 1.4 | 11 | 1.742 | 1.580 | 0.759 | 0.734 | 1.386 | 0.591 | 0.476 | 11 | -0.44 | -0.62 | -0.11 | -0.81 | -1.01 | 0 |
| | 30 | 2 | 1.79 | 0.87 | 0.89 | 1.57 | 10 | 2.419 | 2.126 | 0.962 | 1.153 | 1.896 | 0.764 | 0.584 | 9 | 20.93 | 18.77 | 10.55 | 29.57 | 20.76 | -1 |
| | 45 | 2.77 | 2.41 | 0.92 | 1.36 | 2.23 | 9 | 2.702 | 2.349 | 0.923 | 1.335 | 2.160 | 0.675 | 0.629 | 9 | -2.46 | -2.53 | 0.34 | -1.83 | -3.13 | 0 |
| | 60 | 1.81 | 1.63 | 0.77 | 0.8 | 1.43 | 11 | 1.810 | 1.623 | 0.773 | 0.801 | 1.427 | 0.532 | 0.561 | 11 | -0.02 | -0.46 | 0.39 | 0.17 | -0.24 | 0 |
| | 75 | 2.03 | 1.79 | 0.93 | 0.95 | 1.53 | 10 | 2.041 | 1.799 | 0.922 | 0.964 | 1.545 | 0.572 | 0.723 | 10 | 0.52 | 0.49 | -0.87 | 1.43 | 0.95 | 0 |
| | 90 | 1.84 | 1.62 | 0.93 | 0.87 | 1.33 | 10 | 1.853 | 1.633 | 0.933 | 0.876 | 1.340 | 0.582 | 0.729 | 10 | 0.69 | 0.77 | 0.32 | 0.70 | 0.72 | 0 |
| | 105 | 1.53 | 1.38 | 0.76 | 0.67 | 1.14 | 12 | 1.633 | 1.463 | 0.847 | 0.725 | 1.193 | 0.552 | 0.643 | 11 | 6.75 | 6.04 | 11.45 | 8.22 | 4.68 | -1 |
| | 120 | 2.13 | 1.87 | 0.9 | 1.01 | 1.64 | 10 | 2.213 | 1.934 | 0.950 | 1.076 | 1.684 | 0.742 | 0.594 | 9 | 3.89 | 3.40 | 5.58 | 6.53 | 2.69 | -1 |
| | 135 | 1.79 | 1.59 | 0.81 | 0.83 | 1.37 | 10 | 1.791 | 1.591 | 0.810 | 0.822 | 1.369 | 0.557 | 0.589 | 10 | 0.03 | 0.04 | 0.05 | -0.94 | -0.10 | 0 |
| | 150 | 1.69 | 1.52 | 0.72 | 0.74 | 1.34 | 12 | 1.712 | 1.539 | 0.725 | 0.751 | 1.357 | 0.530 | 0.495 | 12 | 1.33 | 1.23 | 0.71 | 1.54 | 1.28 | 0 |
| | 165 | 2.01 | 1.78 | 0.78 | 0.93 | 1.6 | 11 | 2.034 | 1.802 | 0.785 | 0.944 | 1.622 | 0.521 | 0.588 | 11 | 1.19 | 1.22 | 0.69 | 1.47 | 1.35 | 0 |
| | 180 | 3.04 | 2.62 | 1.06 | 1.56 | 2.39 | 9 | 3.039 | 2.611 | 1.062 | 1.554 | 2.386 | 0.638 | 0.850 | 9 | -0.04 | -0.33 | 0.23 | -0.38 | -0.18 | 0 |

| Location | | Trimble | | | | | | Matlab | | | | | | | | Percentage Difference | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Lat (+/-90) (+North/-South) | Long (+/-180) (+East/-West) | GDOP | PDOP | HDOP | TDOP | VDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | Difference in Visible SV |
| 0 | -165 | 2.12 | 1.84 | 0.94 | 1.05 | 1.58 | 9 | 1.671 | 1.482 | 0.826 | 0.772 | 1.230 | 0.565 | 0.603 | 10 | -21.18 | -19.45 | -12.12 | -26.50 | -22.13 | 1 |
| | -150 | 2.11 | 1.83 | 0.98 | 1.04 | 1.54 | 9 | 2.092 | 1.817 | 0.978 | 1.037 | 1.532 | 0.684 | 0.700 | 9 | -0.84 | -0.69 | -0.16 | -0.32 | -0.55 | 0 |
| | -135 | 1.86 | 1.65 | 0.86 | 0.85 | 1.41 | 9 | 1.878 | 1.666 | 0.869 | 0.866 | 1.422 | 0.628 | 0.600 | 9 | 0.96 | 0.98 | 1.02 | 1.92 | 0.83 | 0 |
| | -120 | 2.35 | 2.06 | 0.92 | 1.13 | 1.85 | 8 | 2.355 | 2.062 | 0.920 | 1.137 | 1.845 | 0.630 | 0.671 | 8 | 0.20 | 0.08 | 0.02 | 0.65 | -0.27 | 0 |
| | -105 | 1.81 | 1.64 | 0.76 | 0.77 | 1.46 | 10 | 1.820 | 1.648 | 0.757 | 0.774 | 1.463 | 0.559 | 0.511 | 10 | 0.57 | 0.47 | -0.36 | 0.48 | 0.23 | 0 |
| | -90 | 1.86 | 1.68 | 0.76 | 0.81 | 1.49 | 11 | 1.846 | 1.662 | 0.758 | 0.804 | 1.479 | 0.542 | 0.530 | 11 | -0.74 | -1.08 | -0.26 | -0.73 | -0.74 | 0 |
| | -75 | 2.02 | 1.79 | 0.8 | 0.93 | 1.61 | 11 | 2.032 | 1.803 | 0.804 | 0.938 | 1.614 | 0.512 | 0.620 | 11 | 0.60 | 0.70 | 0.46 | 0.90 | 0.22 | 0 |
| | -60 | 2.13 | 1.88 | 0.96 | 1.01 | 1.62 | 10 | 2.146 | 1.887 | 0.954 | 1.022 | 1.628 | 0.519 | 0.801 | 10 | 0.77 | 0.39 | -0.60 | 1.20 | 0.51 | 0 |
| | -45 | 1.64 | 1.46 | 0.85 | 0.76 | 1.18 | 11 | 1.469 | 1.315 | 0.759 | 0.655 | 1.074 | 0.468 | 0.597 | 12 | -10.43 | -9.92 | -10.74 | -13.88 | -8.96 | 1 |
| | -30 | 1.82 | 1.6 | 0.83 | 0.88 | 1.37 | 11 | 1.822 | 1.600 | 0.820 | 0.871 | 1.374 | 0.529 | 0.626 | 11 | 0.09 | -0.01 | -1.25 | -0.99 | 0.28 | 0 |
| | -15 | 1.86 | 1.63 | 0.85 | 0.9 | 1.39 | 11 | 1.856 | 1.628 | 0.846 | 0.891 | 1.391 | 0.535 | 0.656 | 11 | -0.23 | -0.12 | -0.42 | -1.04 | 0.06 | 0 |
| | 0 | 1.88 | 1.66 | 0.87 | 0.89 | 1.41 | 10 | 1.858 | 1.640 | 0.863 | 0.873 | 1.395 | 0.559 | 0.657 | 10 | -1.16 | -1.20 | -0.84 | -1.87 | -1.08 | 0 |
| | 15 | 1.61 | 1.46 | 0.73 | 0.67 | 1.27 | 11 | 1.594 | 1.449 | 0.728 | 0.665 | 1.253 | 0.560 | 0.465 | 11 | -1.00 | -0.79 | -0.34 | -0.73 | -1.38 | 0 |
| | 30 | 1.96 | 1.76 | 0.8 | 0.86 | 1.56 | 10 | 1.919 | 1.723 | 0.806 | 0.844 | 1.523 | 0.649 | 0.477 | 10 | -2.11 | -2.09 | 0.70 | -1.90 | -2.35 | 0 |
| | 45 | 2.79 | 2.43 | 0.94 | 1.38 | 2.24 | 9 | 2.058 | 1.815 | 0.838 | 0.971 | 1.610 | 0.555 | 0.628 | 10 | -26.23 | -25.31 | -10.86 | -29.65 | -28.13 | 1 |
| | 60 | 1.75 | 1.57 | 0.77 | 0.79 | 1.37 | 11 | 1.769 | 1.578 | 0.766 | 0.799 | 1.380 | 0.534 | 0.549 | 11 | 1.09 | 0.53 | -0.53 | 1.15 | 0.74 | 0 |
| | 75 | 2.12 | 1.85 | 0.96 | 1.04 | 1.58 | 9 | 2.191 | 1.903 | 0.958 | 1.086 | 1.644 | 0.579 | 0.763 | 9 | 3.36 | 2.86 | -0.19 | 4.46 | 4.06 | 0 |
| | 90 | 1.69 | 1.51 | 0.86 | 0.77 | 1.24 | 11 | 1.592 | 1.422 | 0.806 | 0.717 | 1.171 | 0.531 | 0.607 | 11 | -5.79 | -5.85 | -6.23 | -6.92 | -5.57 | 0 |
| | 105 | 2.15 | 1.9 | 0.86 | 1.01 | 1.69 | 11 | 2.127 | 1.881 | 0.857 | 0.993 | 1.674 | 0.542 | 0.663 | 11 | -1.08 | -1.02 | -0.41 | -1.65 | -0.93 | 0 |
| | 120 | 1.85 | 1.66 | 0.72 | 0.83 | 1.49 | 12 | 1.828 | 1.640 | 0.720 | 0.808 | 1.473 | 0.476 | 0.540 | 12 | -1.19 | -1.22 | -0.04 | -2.69 | -1.12 | 0 |
| | 135 | 1.75 | 1.57 | 0.73 | 0.76 | 1.39 | 12 | 1.825 | 1.639 | 0.766 | 0.803 | 1.448 | 0.493 | 0.586 | 11 | 4.27 | 4.36 | 4.93 | 5.67 | 4.19 | -1 |
| | 150 | 1.77 | 1.58 | 0.78 | 0.78 | 1.38 | 11 | 1.785 | 1.601 | 0.780 | 0.791 | 1.398 | 0.580 | 0.521 | 11 | 0.86 | 1.30 | 0.01 | 1.38 | 1.28 | 0 |
| | 165 | 1.98 | 1.75 | 0.86 | 0.92 | 1.53 | 11 | 2.007 | 1.776 | 0.859 | 0.935 | 1.555 | 0.621 | 0.593 | 11 | 1.36 | 1.50 | -0.15 | 1.59 | 1.62 | 0 |
| | 180 | 1.92 | 1.68 | 0.91 | 0.93 | 1.41 | 9 | 1.935 | 1.691 | 0.912 | 0.942 | 1.424 | 0.579 | 0.705 | 9 | 0.79 | 0.64 | 0.20 | 1.25 | 0.98 | 0 |

| Location | | Trimble | | | | | | Matlab | | | | | | | | Percentage Difference | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Lat (+/-90) (+North/-South) | Long (+/-180) (+East/-West) | GDOP | PDOP | HDOP | TDOP | VDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | Difference in Visible SV |
| 15 | -165 | 2.07 | 1.79 | 1.04 | 1.04 | 1.46 | 8 | 2.076 | 1.795 | 1.041 | 1.043 | 1.462 | 0.726 | 0.746 | 8 | 0.29 | 0.28 | 0.09 | 0.28 | 0.16 | 0 |
| | -150 | 1.52 | 1.36 | 0.81 | 0.68 | 1.1 | 10 | 1.527 | 1.367 | 0.813 | 0.680 | 1.098 | 0.576 | 0.574 | 10 | 0.44 | 0.50 | 0.42 | 0.01 | -0.15 | 0 |
| | -135 | 2.13 | 1.87 | 0.98 | 1.01 | 1.59 | 8 | 2.174 | 1.909 | 0.989 | 1.041 | 1.633 | 0.759 | 0.634 | 8 | 2.06 | 2.07 | 0.88 | 3.02 | 2.69 | 0 |
| | -120 | 2.9 | 2.53 | 0.95 | 1.43 | 2.34 | 8 | 2.957 | 2.572 | 0.958 | 1.460 | 2.387 | 0.695 | 0.659 | 8 | 1.98 | 1.64 | 0.81 | 2.13 | 1.99 | 0 |
| | -105 | 2.94 | 2.56 | 0.97 | 1.45 | 2.37 | 8 | 2.997 | 2.605 | 0.981 | 1.481 | 2.414 | 0.692 | 0.695 | 8 | 1.93 | 1.77 | 1.13 | 2.12 | 1.84 | 0 |
| | -90 | 2.03 | 1.81 | 0.81 | 0.91 | 1.63 | 10 | 1.906 | 1.712 | 0.776 | 0.839 | 1.526 | 0.565 | 0.532 | 11 | -6.12 | -5.44 | -4.21 | -7.86 | -6.41 | 1 |
| | -75 | 2.54 | 2.22 | 0.84 | 1.23 | 2.06 | 10 | 2.580 | 2.255 | 0.837 | 1.253 | 2.094 | 0.562 | 0.620 | 10 | 1.56 | 1.57 | -0.39 | 1.89 | 1.64 | 0 |
| | -60 | 1.74 | 1.54 | 0.77 | 0.79 | 1.34 | 11 | 1.753 | 1.557 | 0.765 | 0.805 | 1.356 | 0.516 | 0.565 | 11 | 0.72 | 1.09 | -0.66 | 1.86 | 1.19 | 0 |
| | -45 | 1.88 | 1.64 | 0.87 | 0.92 | 1.39 | 10 | 2.272 | 1.956 | 0.960 | 1.154 | 1.705 | 0.567 | 0.774 | 9 | 20.83 | 19.29 | 10.31 | 25.48 | 22.65 | -1 |
| | -30 | 1.67 | 1.48 | 0.79 | 0.79 | 1.25 | 11 | 1.671 | 1.476 | 0.784 | 0.784 | 1.251 | 0.535 | 0.573 | 11 | 0.07 | -0.26 | -0.75 | -0.81 | 0.05 | 0 |
| | -15 | 2.37 | 2.03 | 1.01 | 1.22 | 1.76 | 9 | 2.354 | 2.020 | 0.998 | 1.208 | 1.756 | 0.709 | 0.703 | 9 | -0.70 | -0.49 | -1.15 | -1.02 | -0.23 | 0 |
| | 0 | 1.92 | 1.69 | 0.9 | 0.91 | 1.42 | 9 | 1.685 | 1.512 | 0.839 | 0.745 | 1.258 | 0.654 | 0.526 | 10 | -12.23 | -10.55 | -6.77 | -18.18 | -11.44 | 1 |
| | 15 | 1.87 | 1.67 | 0.87 | 0.85 | 1.42 | 9 | 1.869 | 1.665 | 0.867 | 0.851 | 1.421 | 0.653 | 0.570 | 9 | -0.03 | -0.32 | -0.38 | 0.06 | 0.09 | 0 |
| | 30 | 1.55 | 1.41 | 0.74 | 0.66 | 1.2 | 11 | 1.549 | 1.402 | 0.740 | 0.658 | 1.191 | 0.547 | 0.498 | 11 | -0.10 | -0.58 | -0.07 | -0.32 | -0.77 | 0 |
| | 45 | 2.3 | 2.01 | 1 | 1.12 | 1.74 | 9 | 2.296 | 2.003 | 1.005 | 1.122 | 1.733 | 0.749 | 0.670 | 9 | -0.17 | -0.33 | 0.45 | 0.19 | -0.39 | 0 |
| | 60 | 2.28 | 1.98 | 0.99 | 1.14 | 1.72 | 9 | 2.299 | 1.991 | 0.987 | 1.150 | 1.730 | 0.736 | 0.657 | 9 | 0.84 | 0.57 | -0.34 | 0.84 | 0.56 | 0 |
| | 75 | 1.61 | 1.43 | 0.83 | 0.74 | 1.16 | 10 | 1.492 | 1.336 | 0.781 | 0.664 | 1.083 | 0.529 | 0.576 | 11 | -7.34 | -6.59 | -5.86 | -10.26 | -6.61 | 1 |
| | 90 | 1.75 | 1.55 | 0.85 | 0.82 | 1.29 | 10 | 1.788 | 1.578 | 0.844 | 0.841 | 1.333 | 0.577 | 0.616 | 10 | 2.18 | 1.80 | -0.73 | 2.60 | 3.36 | 0 |
| | 105 | 3.03 | 2.62 | 0.92 | 1.51 | 2.45 | 10 | 3.005 | 2.610 | 0.913 | 1.489 | 2.445 | 0.568 | 0.715 | 10 | -0.84 | -0.39 | -0.75 | -1.41 | -0.20 | 0 |
| | 120 | 2.16 | 1.91 | 0.74 | 0.99 | 1.76 | 11 | 2.157 | 1.918 | 0.740 | 0.988 | 1.769 | 0.549 | 0.495 | 11 | -0.13 | 0.40 | -0.07 | -0.18 | 0.52 | 0 |
| | 135 | 2.33 | 2.05 | 0.9 | 1.11 | 1.84 | 10 | 2.352 | 2.072 | 0.905 | 1.113 | 1.864 | 0.597 | 0.680 | 10 | 0.92 | 1.05 | 0.52 | 0.25 | 1.28 | 0 |
| | 150 | 1.66 | 1.49 | 0.78 | 0.72 | 1.27 | 11 | 1.658 | 1.493 | 0.780 | 0.721 | 1.273 | 0.606 | 0.492 | 11 | -0.13 | 0.21 | 0.04 | 0.08 | 0.24 | 0 |
| | 165 | 1.94 | 1.72 | 1.05 | 0.9 | 1.36 | 8 | 1.932 | 1.712 | 1.052 | 0.894 | 1.351 | 0.713 | 0.773 | 8 | -0.43 | -0.45 | 0.15 | -0.66 | -0.65 | 0 |
| | 180 | 2.08 | 1.8 | 1.01 | 1.05 | 1.49 | 8 | 2.098 | 1.812 | 1.015 | 1.058 | 1.501 | 0.730 | 0.705 | 8 | 0.87 | 0.65 | 0.47 | 0.77 | 0.73 | 0 |

| Location | | Trimble | | | | | | Matlab | | | | | | | | Percentage Difference | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Lat (+/-90) (+North/-South) | Long (+/-180) (+East/-West) | GDOP | PDOP | HDOP | TDOP | VDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | Difference in Visible SV |
| 30 | -165 | 2.48 | 2.14 | 1.05 | 1.25 | 1.86 | 8 | 2.472 | 2.132 | 1.053 | 1.251 | 1.854 | 0.830 | 0.648 | 8 | -0.33 | -0.38 | 0.28 | 0.10 | -0.34 | 0 |
| | -150 | 2.54 | 2.19 | 1.18 | 1.29 | 1.84 | 8 | 2.531 | 2.179 | 1.171 | 1.288 | 1.837 | 0.868 | 0.786 | 8 | -0.37 | -0.52 | -0.79 | -0.19 | -0.15 | 0 |
| | -135 | 1.73 | 1.55 | 0.86 | 0.76 | 1.29 | 9 | 1.763 | 1.576 | 0.860 | 0.789 | 1.321 | 0.657 | 0.555 | 9 | 1.90 | 1.70 | 0.03 | 3.84 | 2.39 | 0 |
| | -120 | 2.09 | 1.86 | 0.93 | 0.95 | 1.61 | 9 | 2.105 | 1.873 | 0.937 | 0.960 | 1.622 | 0.673 | 0.652 | 9 | 0.69 | 0.70 | 0.78 | 1.01 | 0.72 | 0 |
| | -105 | 2.08 | 1.85 | 0.88 | 0.94 | 1.63 | 9 | 2.091 | 1.864 | 0.888 | 0.950 | 1.638 | 0.687 | 0.563 | 9 | 0.55 | 0.73 | 0.92 | 1.01 | 0.50 | 0 |
| | -90 | 2.06 | 1.84 | 0.98 | 0.94 | 1.55 | 10 | 2.056 | 1.831 | 0.980 | 0.937 | 1.546 | 0.817 | 0.541 | 10 | -0.18 | -0.51 | -0.01 | -0.37 | -0.23 | 0 |
| | -75 | 2.99 | 2.58 | 1.15 | 1.51 | 2.31 | 9 | 3.032 | 2.614 | 1.154 | 1.536 | 2.345 | 0.969 | 0.627 | 9 | 1.39 | 1.31 | 0.34 | 1.71 | 1.53 | 0 |
| | -60 | 2.2 | 1.91 | 0.92 | 1.1 | 1.67 | 9 | 2.208 | 1.914 | 0.920 | 1.102 | 1.678 | 0.671 | 0.630 | 9 | 0.37 | 0.20 | 0.04 | 0.15 | 0.47 | 0 |
| | -45 | 1.72 | 1.52 | 0.83 | 0.8 | 1.27 | 10 | 1.697 | 1.503 | 0.832 | 0.788 | 1.252 | 0.631 | 0.542 | 10 | -1.33 | -1.11 | 0.22 | -1.50 | -1.41 | 0 |
| | -30 | 2.16 | 1.86 | 0.95 | 1.1 | 1.59 | 9 | 2.113 | 1.825 | 0.947 | 1.065 | 1.561 | 0.713 | 0.623 | 9 | -2.18 | -1.88 | -0.37 | -3.23 | -1.86 | 0 |
| | -15 | 1.93 | 1.7 | 0.91 | 0.92 | 1.43 | 9 | 1.905 | 1.676 | 0.898 | 0.904 | 1.415 | 0.685 | 0.581 | 9 | -1.31 | -1.39 | -1.29 | -1.71 | -1.03 | 0 |
| | 0 | 2.34 | 2.06 | 1.17 | 1.11 | 1.69 | 8 | 2.391 | 2.100 | 1.153 | 1.143 | 1.755 | 1.007 | 0.561 | 8 | 2.17 | 1.94 | -1.45 | 2.95 | 3.85 | 0 |
| | 15 | 2.14 | 1.91 | 1.16 | 0.97 | 1.52 | 9 | 2.188 | 1.947 | 1.150 | 1.000 | 1.571 | 1.015 | 0.540 | 9 | 2.26 | 1.92 | -0.87 | 3.06 | 3.34 | 0 |
| | 30 | 1.79 | 1.6 | 0.98 | 0.79 | 1.27 | 9 | 1.782 | 1.596 | 0.972 | 0.793 | 1.266 | 0.820 | 0.522 | 9 | -0.44 | -0.24 | -0.81 | 0.37 | -0.31 | 0 |
| | 45 | 2.14 | 1.87 | 1.06 | 1.05 | 1.54 | 8 | 2.111 | 1.842 | 1.049 | 1.031 | 1.514 | 0.818 | 0.657 | 8 | -1.37 | -1.50 | -1.03 | -1.85 | -1.69 | 0 |
| | 60 | 2.13 | 1.84 | 1.03 | 1.07 | 1.53 | 8 | 2.097 | 1.813 | 1.029 | 1.054 | 1.493 | 0.771 | 0.681 | 8 | -1.54 | -1.46 | -0.09 | -1.50 | -2.43 | 0 |
| | 75 | 1.82 | 1.59 | 0.93 | 0.88 | 1.29 | 9 | 1.808 | 1.583 | 0.925 | 0.874 | 1.284 | 0.687 | 0.619 | 9 | -0.68 | -0.47 | -0.53 | -0.72 | -0.47 | 0 |
| | 90 | 1.89 | 1.65 | 0.96 | 0.91 | 1.34 | 9 | 1.915 | 1.675 | 0.965 | 0.928 | 1.369 | 0.741 | 0.618 | 9 | 1.31 | 1.51 | 0.54 | 1.97 | 2.14 | 0 |
| | 105 | 3.44 | 2.95 | 1.28 | 1.77 | 2.66 | 9 | 3.370 | 2.898 | 1.243 | 1.719 | 2.618 | 1.003 | 0.733 | 9 | -2.05 | -1.76 | -2.92 | -2.89 | -1.57 | 0 |
| | 120 | 2.29 | 2.02 | 0.92 | 1.08 | 1.8 | 10 | 2.683 | 2.341 | 0.929 | 1.309 | 2.149 | 0.764 | 0.529 | 9 | 17.14 | 15.91 | 1.00 | 21.21 | 19.39 | -1 |
| | 135 | 2.66 | 2.31 | 1.04 | 1.31 | 2.07 | 9 | 2.662 | 2.322 | 1.027 | 1.303 | 2.082 | 0.756 | 0.695 | 9 | 0.09 | 0.50 | -1.24 | -0.52 | 0.58 | 0 |
| | 150 | 2.4 | 2.1 | 1.22 | 1.17 | 1.71 | 7 | 2.361 | 2.068 | 1.208 | 1.138 | 1.679 | 1.004 | 0.671 | 7 | -1.64 | -1.50 | -1.01 | -2.74 | -1.80 | 0 |
| | 165 | 2.34 | 2.03 | 1.15 | 1.15 | 1.67 | 8 | 2.294 | 1.997 | 1.148 | 1.129 | 1.634 | 0.885 | 0.731 | 8 | -1.97 | -1.63 | -0.18 | -1.86 | -2.15 | 0 |
| | 180 | 2.33 | 2.03 | 1.08 | 1.15 | 1.71 | 8 | 2.287 | 1.993 | 1.082 | 1.122 | 1.674 | 0.853 | 0.665 | 8 | -1.83 | -1.81 | 0.16 | -2.41 | -2.10 | 0 |

| Location | | Trimble | | | | | | Matlab | | | | | | | | Percentage Difference | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Lat (+/-90) (+North/-South) | Long (+/-180) (+East/-West) | GDOP | PDOP | HDOP | TDOP | VDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | Difference in Visible SV |
| 45 | -165 | 2.15 | 1.89 | 0.89 | 1.01 | 1.67 | 9 | 2.119 | 1.869 | 0.886 | 0.999 | 1.645 | 0.638 | 0.614 | 9 | -1.45 | -1.14 | -0.48 | -1.09 | -1.49 | 0 |
| | -150 | 3.45 | 2.91 | 1.14 | 1.84 | 2.68 | 7 | 3.540 | 2.983 | 1.137 | 1.906 | 2.758 | 0.826 | 0.781 | 7 | 2.59 | 2.50 | -0.29 | 3.57 | 2.90 | 0 |
| | -135 | 1.98 | 1.78 | 0.97 | 0.85 | 1.5 | 9 | 2.009 | 1.808 | 0.972 | 0.875 | 1.525 | 0.771 | 0.592 | 9 | 1.44 | 1.58 | 0.19 | 2.91 | 1.65 | 0 |
| | -120 | 2.1 | 1.88 | 1.07 | 0.94 | 1.55 | 8 | 2.120 | 1.895 | 1.076 | 0.950 | 1.560 | 0.846 | 0.665 | 8 | 0.94 | 0.81 | 0.53 | 1.02 | 0.66 | 0 |
| | -105 | 2.15 | 1.92 | 1.17 | 0.97 | 1.52 | 8 | 2.162 | 1.927 | 1.175 | 0.981 | 1.527 | 1.021 | 0.581 | 8 | 0.55 | 0.34 | 0.40 | 1.15 | 0.45 | 0 |
| | -90 | 2.04 | 1.82 | 1.2 | 0.91 | 1.37 | 7 | 2.042 | 1.824 | 1.211 | 0.918 | 1.364 | 1.021 | 0.652 | 7 | 0.10 | 0.23 | 0.92 | 0.89 | -0.42 | 0 |
| | -75 | 3.76 | 3.16 | 1.81 | 2.04 | 2.6 | 7 | 3.794 | 3.187 | 1.817 | 2.058 | 2.617 | 1.638 | 0.786 | 7 | 0.89 | 0.84 | 0.40 | 0.90 | 0.67 | 0 |
| | -60 | 2.23 | 1.92 | 1.12 | 1.12 | 1.56 | 9 | 2.230 | 1.926 | 1.120 | 1.125 | 1.566 | 0.925 | 0.631 | 9 | 0.01 | 0.30 | 0.02 | 0.46 | 0.40 | 0 |
| | -45 | 1.74 | 1.54 | 0.93 | 0.82 | 1.23 | 10 | 2.226 | 1.920 | 1.145 | 1.127 | 1.541 | 0.887 | 0.724 | 9 | 27.91 | 24.64 | 23.14 | 37.39 | 25.24 | -1 |
| | -30 | 2.66 | 2.25 | 1.22 | 1.42 | 1.89 | 8 | 3.197 | 2.692 | 1.293 | 1.724 | 2.362 | 1.002 | 0.816 | 7 | 20.19 | 19.66 | 5.94 | 21.42 | 24.96 | -1 |
| | -15 | 2.52 | 2.19 | 0.99 | 1.26 | 1.95 | 8 | 2.495 | 2.168 | 0.987 | 1.234 | 1.930 | 0.768 | 0.620 | 8 | -1.00 | -1.00 | -0.29 | -2.06 | -1.01 | 0 |
| | 0 | 1.98 | 1.78 | 0.86 | 0.87 | 1.56 | 10 | 2.020 | 1.810 | 0.853 | 0.897 | 1.597 | 0.703 | 0.483 | 10 | 2.02 | 1.69 | -0.83 | 3.06 | 2.35 | 0 |
| | 15 | 2.01 | 1.8 | 0.9 | 0.89 | 1.56 | 10 | 2.049 | 1.836 | 0.896 | 0.910 | 1.603 | 0.730 | 0.519 | 10 | 1.94 | 1.98 | -0.50 | 2.28 | 2.72 | 0 |
| | 30 | 1.88 | 1.7 | 1.02 | 0.8 | 1.36 | 9 | 1.883 | 1.700 | 1.009 | 0.808 | 1.369 | 0.881 | 0.492 | 9 | 0.14 | 0.02 | -1.07 | 1.05 | 0.63 | 0 |
| | 45 | 2.48 | 2.16 | 1.08 | 1.22 | 1.88 | 8 | 2.407 | 2.100 | 1.067 | 1.175 | 1.809 | 0.872 | 0.615 | 8 | -2.96 | -2.77 | -1.17 | -3.68 | -3.80 | 0 |
| | 60 | 1.82 | 1.6 | 0.89 | 0.85 | 1.33 | 9 | 2.358 | 2.058 | 1.065 | 1.150 | 1.761 | 0.867 | 0.619 | 8 | 29.55 | 28.63 | 19.66 | 35.34 | 32.41 | -1 |
| | 75 | 1.82 | 1.6 | 0.95 | 0.86 | 1.3 | 9 | 1.785 | 1.575 | 0.942 | 0.840 | 1.263 | 0.647 | 0.684 | 9 | -1.93 | -1.56 | -0.86 | -2.36 | -2.88 | 0 |
| | 90 | 2.16 | 1.86 | 1.2 | 1.1 | 1.42 | 8 | 2.603 | 2.246 | 1.446 | 1.316 | 1.718 | 1.112 | 0.925 | 7 | 20.51 | 20.75 | 20.53 | 19.64 | 21.00 | -1 |
| | 105 | 2.93 | 2.54 | 1.47 | 1.47 | 2.07 | 8 | 2.948 | 2.553 | 1.463 | 1.474 | 2.092 | 1.165 | 0.884 | 8 | 0.60 | 0.50 | -0.48 | 0.27 | 1.05 | 0 |
| | 120 | 2.87 | 2.49 | 1.32 | 1.43 | 2.11 | 8 | 2.885 | 2.506 | 1.311 | 1.431 | 2.136 | 1.193 | 0.543 | 8 | 0.54 | 0.63 | -0.68 | 0.05 | 1.21 | 0 |
| | 135 | 2.86 | 2.48 | 1.36 | 1.43 | 2.07 | 8 | 2.870 | 2.492 | 1.346 | 1.425 | 2.097 | 1.154 | 0.692 | 8 | 0.36 | 0.46 | -1.06 | -0.34 | 1.29 | 0 |
| | 150 | 1.67 | 1.52 | 0.94 | 0.69 | 1.2 | 9 | 1.657 | 1.509 | 0.935 | 0.685 | 1.185 | 0.788 | 0.503 | 9 | -0.76 | -0.72 | -0.57 | -0.71 | -1.26 | 0 |
| | 165 | 2.14 | 1.88 | 0.98 | 1.02 | 1.61 | 9 | 2.109 | 1.858 | 0.977 | 1.000 | 1.580 | 0.657 | 0.724 | 9 | -1.43 | -1.20 | -0.28 | -1.99 | -1.89 | 0 |
| | 180 | 2.13 | 1.88 | 0.88 | 1 | 1.66 | 9 | 2.098 | 1.851 | 0.877 | 0.988 | 1.630 | 0.642 | 0.597 | 9 | -1.52 | -1.56 | -0.39 | -1.25 | -1.82 | 0 |

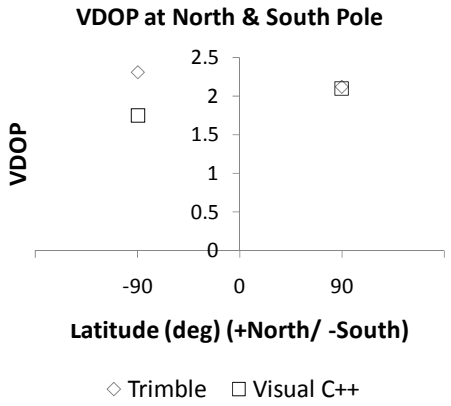
| Location | | Trimble | | | | | | Matlab | | | | | | | | Percentage Difference | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Lat (+/-90) (+North/-South) | Long (+/-180) (+East/-West) | GDOP | PDOP | HDOP | TDOP | VDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | Difference in Visible SV |
| 60 | -165 | 2.04 | 1.83 | 0.83 | 0.89 | 1.64 | 9 | 2.041 | 1.833 | 0.833 | 0.898 | 1.632 | 0.544 | 0.631 | 9 | 0.02 | 0.14 | 0.35 | 0.84 | -0.47 | 0 |
| | -150 | 1.85 | 1.69 | 0.77 | 0.77 | 1.5 | 10 | 1.872 | 1.699 | 0.767 | 0.786 | 1.515 | 0.544 | 0.541 | 10 | 1.17 | 0.51 | -0.36 | 2.09 | 1.03 | 0 |
| | -135 | 1.85 | 1.68 | 0.8 | 0.77 | 1.48 | 10 | 1.871 | 1.697 | 0.799 | 0.787 | 1.498 | 0.575 | 0.554 | 10 | 1.13 | 1.04 | -0.19 | 2.18 | 1.21 | 0 |
| | -120 | 2.33 | 2.08 | 1.01 | 1.04 | 1.82 | 8 | 2.303 | 2.058 | 1.005 | 1.034 | 1.796 | 0.759 | 0.659 | 8 | -1.16 | -1.07 | -0.48 | -0.57 | -1.34 | 0 |
| | -105 | 4.21 | 3.66 | 1.77 | 2.07 | 3.21 | 6 | 4.315 | 3.751 | 1.810 | 2.134 | 3.285 | 1.624 | 0.799 | 6 | 2.50 | 2.48 | 2.26 | 3.09 | 2.35 | 0 |
| | -90 | 4.2 | 3.65 | 1.9 | 2.07 | 3.11 | 6 | 4.312 | 3.742 | 1.952 | 2.143 | 3.192 | 1.691 | 0.976 | 6 | 2.66 | 2.51 | 2.75 | 3.51 | 2.64 | 0 |
| | -75 | 2.08 | 1.94 | 1.48 | 0.75 | 1.26 | 7 | 2.100 | 1.960 | 1.503 | 0.753 | 1.258 | 1.381 | 0.594 | 7 | 0.95 | 1.03 | 1.55 | 0.45 | -0.16 | 0 |
| | -60 | 2.57 | 2.23 | 1.54 | 1.28 | 1.61 | 7 | 2.578 | 2.234 | 1.549 | 1.287 | 1.609 | 1.360 | 0.741 | 7 | 0.31 | 0.16 | 0.59 | 0.58 | -0.07 | 0 |
| | -45 | 1.76 | 1.59 | 0.88 | 0.75 | 1.32 | 9 | 1.836 | 1.652 | 0.955 | 0.802 | 1.348 | 0.654 | 0.696 | 8 | 4.32 | 3.88 | 8.52 | 6.92 | 2.10 | -1 |
| | -30 | 1.72 | 1.56 | 0.8 | 0.72 | 1.34 | 10 | 1.711 | 1.553 | 0.800 | 0.719 | 1.331 | 0.586 | 0.544 | 10 | -0.51 | -0.44 | -0.04 | -0.17 | -0.65 | 0 |
| | -15 | 2.19 | 1.94 | 0.91 | 1.01 | 1.71 | 9 | 2.197 | 1.948 | 0.919 | 1.015 | 1.718 | 0.679 | 0.619 | 9 | 0.31 | 0.43 | 0.99 | 0.46 | 0.47 | 0 |
| | 0 | 2.01 | 1.81 | 0.85 | 0.89 | 1.6 | 10 | 2.438 | 2.167 | 0.892 | 1.117 | 1.975 | 0.721 | 0.526 | 9 | 21.27 | 19.71 | 4.95 | 25.45 | 23.41 | -1 |
| | 15 | 2.09 | 1.89 | 0.79 | 0.89 | 1.72 | 10 | 2.116 | 1.910 | 0.785 | 0.910 | 1.741 | 0.618 | 0.484 | 10 | 1.22 | 1.06 | -0.63 | 2.19 | 1.24 | 0 |
| | 30 | 2.77 | 2.47 | 0.93 | 1.25 | 2.29 | 9 | 2.758 | 2.459 | 0.930 | 1.248 | 2.277 | 0.750 | 0.550 | 9 | -0.44 | -0.43 | -0.03 | -0.14 | -0.58 | 0 |
| | 45 | 2.11 | 1.89 | 0.85 | 0.93 | 1.68 | 9 | 2.063 | 1.849 | 0.853 | 0.914 | 1.641 | 0.630 | 0.575 | 9 | -2.23 | -2.15 | 0.34 | -1.71 | -2.33 | 0 |
| | 60 | 2.48 | 2.16 | 1 | 1.21 | 1.92 | 8 | 2.402 | 2.100 | 1.000 | 1.167 | 1.847 | 0.783 | 0.622 | 8 | -3.14 | -2.79 | -0.05 | -3.59 | -3.82 | 0 |
| | 75 | 2.47 | 2.15 | 1.08 | 1.21 | 1.86 | 8 | 2.395 | 2.091 | 1.076 | 1.168 | 1.793 | 0.770 | 0.751 | 8 | -3.04 | -2.75 | -0.42 | -3.46 | -3.60 | 0 |
| | 90 | 2.49 | 2.16 | 1.22 | 1.23 | 1.79 | 8 | 2.515 | 2.188 | 1.229 | 1.240 | 1.810 | 0.776 | 0.953 | 7 | 1.00 | 1.30 | 0.76 | 0.80 | 1.12 | -1 |
| | 105 | 1.85 | 1.67 | 0.96 | 0.79 | 1.37 | 8 | 1.857 | 1.679 | 0.960 | 0.793 | 1.378 | 0.760 | 0.587 | 8 | 0.39 | 0.56 | 0.02 | 0.39 | 0.57 | 0 |
| | 120 | 1.69 | 1.53 | 0.87 | 0.71 | 1.26 | 9 | 1.740 | 1.574 | 0.880 | 0.742 | 1.305 | 0.688 | 0.548 | 9 | 2.94 | 2.84 | 1.09 | 4.52 | 3.56 | 0 |
| | 135 | 1.73 | 1.57 | 0.86 | 0.73 | 1.31 | 9 | 1.737 | 1.572 | 0.862 | 0.739 | 1.315 | 0.688 | 0.519 | 9 | 0.39 | 0.11 | 0.19 | 1.22 | 0.34 | 0 |
| | 150 | 1.76 | 1.61 | 0.84 | 0.73 | 1.37 | 9 | 1.777 | 1.618 | 0.838 | 0.733 | 1.385 | 0.668 | 0.506 | 9 | 0.95 | 0.52 | -0.24 | 0.47 | 1.06 | 0 |
| | 165 | 1.85 | 1.68 | 0.83 | 0.79 | 1.45 | 9 | 1.830 | 1.654 | 0.833 | 0.783 | 1.429 | 0.621 | 0.555 | 9 | -1.06 | -1.54 | 0.33 | -0.86 | -1.43 | 0 |
| | 180 | 1.93 | 1.74 | 0.83 | 0.83 | 1.53 | 9 | 1.901 | 1.716 | 0.835 | 0.817 | 1.500 | 0.561 | 0.618 | 9 | -1.52 | -1.36 | 0.59 | -1.63 | -1.99 | 0 |

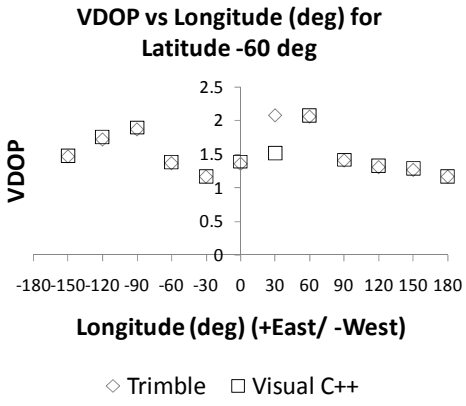
| Location | | Trimble | | | | | | Matlab | | | | | | | | Percentage Difference | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Lat (+/-90) +North/-South) | Long (+/-180) (+East/-West) | GDOP | PDOP | HDOP | TDOP | VDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | Difference in Visible SV |
| 75 | -165 | 2.33 | 2.12 | 0.79 | 0.98 | 1.96 | 9 | 2.401 | 2.173 | 0.798 | 1.020 | 2.021 | 0.600 | 0.525 | 9 | 3.03 | 2.50 | 0.96 | 4.09 | 3.13 | 0 |
| | -150 | 2.34 | 2.12 | 0.81 | 0.99 | 1.96 | 9 | 2.412 | 2.183 | 0.818 | 1.026 | 2.024 | 0.619 | 0.534 | 9 | 3.07 | 2.96 | 0.94 | 3.63 | 3.26 | 0 |
| | -135 | 4.16 | 3.66 | 1.01 | 1.99 | 3.52 | 7 | 4.261 | 3.739 | 1.009 | 2.044 | 3.600 | 0.820 | 0.587 | 7 | 2.43 | 2.15 | -0.13 | 2.71 | 2.28 | 0 |
| | -120 | 4.13 | 3.63 | 1.05 | 1.97 | 3.47 | 7 | 4.227 | 3.708 | 1.055 | 2.029 | 3.555 | 0.817 | 0.667 | 7 | 2.35 | 2.16 | 0.43 | 2.99 | 2.46 | 0 |
| | -105 | 4.11 | 3.61 | 1.13 | 1.96 | 3.42 | 7 | 4.208 | 3.690 | 1.145 | 2.023 | 3.508 | 0.823 | 0.795 | 7 | 2.38 | 2.21 | 1.29 | 3.22 | 2.56 | 0 |
| | -90 | 3.23 | 2.88 | 1.1 | 1.46 | 2.66 | 8 | 3.248 | 2.894 | 1.112 | 1.474 | 2.672 | 0.856 | 0.710 | 8 | 0.54 | 0.48 | 1.08 | 0.93 | 0.44 | 0 |
| | -75 | 2.46 | 2.22 | 0.94 | 1.05 | 2.02 | 9 | 2.451 | 2.216 | 0.945 | 1.047 | 2.005 | 0.767 | 0.552 | 9 | -0.35 | -0.16 | 0.56 | -0.27 | -0.75 | 0 |
| | -60 | 2.45 | 2.22 | 0.94 | 1.04 | 2.01 | 9 | 2.445 | 2.211 | 0.940 | 1.044 | 2.002 | 0.764 | 0.548 | 9 | -0.19 | -0.40 | -0.03 | 0.40 | -0.42 | 0 |
| | -45 | 2.81 | 2.52 | 0.98 | 1.25 | 2.32 | 8 | 2.826 | 2.528 | 0.990 | 1.263 | 2.327 | 0.760 | 0.634 | 8 | 0.58 | 0.33 | 0.99 | 1.02 | 0.29 | 0 |
| | -30 | 2.81 | 2.52 | 0.96 | 1.25 | 2.33 | 8 | 2.830 | 2.532 | 0.959 | 1.263 | 2.344 | 0.758 | 0.587 | 8 | 0.70 | 0.49 | -0.11 | 1.03 | 0.59 | 0 |
| | -15 | 2.27 | 2.05 | 0.81 | 0.96 | 1.89 | 9 | 2.104 | 1.917 | 0.801 | 0.867 | 1.742 | 0.611 | 0.517 | 10 | -7.32 | -6.49 | -1.12 | -9.70 | -7.86 | 1 |
| | 0 | 2.11 | 1.93 | 0.77 | 0.87 | 1.77 | 10 | 2.105 | 1.919 | 0.769 | 0.866 | 1.758 | 0.592 | 0.491 | 10 | -0.24 | -0.59 | -0.10 | -0.46 | -0.69 | 0 |
| | 15 | 2.79 | 2.5 | 0.86 | 1.25 | 2.34 | 9 | 2.779 | 2.481 | 0.860 | 1.251 | 2.328 | 0.676 | 0.532 | 9 | -0.39 | -0.74 | -0.06 | 0.10 | -0.52 | 0 |
| | 30 | 2.8 | 2.5 | 0.85 | 1.25 | 2.35 | 9 | 2.782 | 2.484 | 0.858 | 1.253 | 2.332 | 0.661 | 0.547 | 9 | -0.63 | -0.63 | 0.88 | 0.23 | -0.78 | 0 |
| | 45 | 2.81 | 2.51 | 0.89 | 1.26 | 2.34 | 9 | 2.794 | 2.493 | 0.903 | 1.260 | 2.324 | 0.620 | 0.657 | 9 | -0.59 | -0.68 | 1.51 | 0.03 | -0.71 | 0 |
| | 60 | 2.83 | 2.52 | 0.98 | 1.27 | 2.32 | 9 | 2.812 | 2.507 | 0.988 | 1.274 | 2.304 | 0.573 | 0.805 | 9 | -0.64 | -0.52 | 0.79 | 0.28 | -0.68 | 0 |
| | 75 | 2.38 | 2.14 | 0.89 | 1.02 | 1.95 | 8 | 2.298 | 2.084 | 0.869 | 0.967 | 1.895 | 0.641 | 0.586 | 9 | -3.45 | -2.60 | -2.39 | -5.17 | -2.84 | 1 |
| | 90 | 2.37 | 2.14 | 0.91 | 1.02 | 1.94 | 8 | 2.396 | 2.161 | 0.921 | 1.035 | 1.955 | 0.690 | 0.609 | 8 | 1.09 | 0.97 | 1.16 | 1.49 | 0.76 | 0 |
| | 105 | 2.71 | 2.42 | 0.99 | 1.22 | 2.21 | 7 | 2.771 | 2.471 | 1.001 | 1.254 | 2.259 | 0.727 | 0.688 | 7 | 2.26 | 2.11 | 1.08 | 2.82 | 2.24 | 0 |
| | 120 | 2.07 | 1.88 | 0.83 | 0.86 | 1.69 | 9 | 2.127 | 1.934 | 0.833 | 0.886 | 1.745 | 0.612 | 0.566 | 9 | 2.76 | 2.86 | 0.39 | 3.05 | 3.25 | 0 |
| | 135 | 2.07 | 1.88 | 0.8 | 0.85 | 1.7 | 9 | 2.128 | 1.935 | 0.808 | 0.885 | 1.758 | 0.617 | 0.521 | 9 | 2.78 | 2.91 | 0.95 | 4.11 | 3.42 | 0 |
| | 150 | 2.08 | 1.89 | 0.79 | 0.86 | 1.72 | 9 | 2.135 | 1.942 | 0.796 | 0.888 | 1.771 | 0.617 | 0.503 | 9 | 2.64 | 2.74 | 0.71 | 3.22 | 2.98 | 0 |
| | 165 | 2.62 | 2.35 | 0.85 | 1.16 | 2.19 | 8 | 2.389 | 2.162 | 0.824 | 1.017 | 1.999 | 0.577 | 0.587 | 9 | -8.80 | -8.00 | -3.11 | -12.31 | -8.72 | 1 |
| | 180 | 2.33 | 2.11 | 0.8 | 0.98 | 1.95 | 9 | 2.393 | 2.166 | 0.800 | 1.017 | 2.013 | 0.585 | 0.546 | 9 | 2.70 | 2.65 | -0.01 | 3.79 | 3.22 | 0 |
| 90 | 0 | 2.48 | 2.24 | 0.77 | 1.04 | 2.11 | 9 | 2.465 | 2.233 | 0.775 | 1.043 | 2.095 | 0.552 | 0.544 | 9 | -0.60 | -0.29 | 0.68 | 0.31 | -0.73 | 0 |
| -90 | 0 | 2.74 | 2.44 | 0.83 | 1.26 | 2.29 | 9 | 2.106 | 1.901 | 0.759 | 0.906 | 1.743 | 0.559 | 0.514 | 10 | -23.14 | -22.08 | -8.54 | -28.10 | -23.88 | 1 |

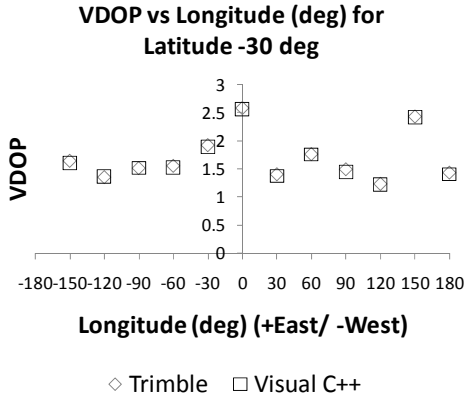
THIS PAGE INTENTIONALLY LEFT BLANK

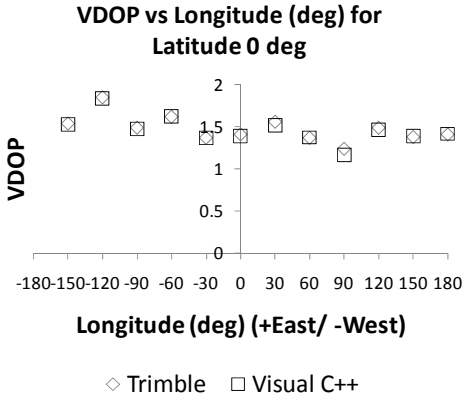
# APPENDIX E. DOP COMPARISON BETWEEN VISUAL C++ & TRIMBLE

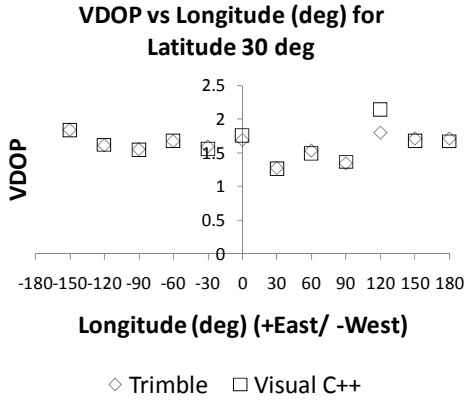
## A. VDOP

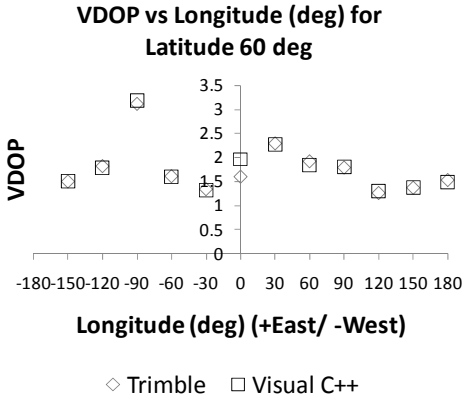


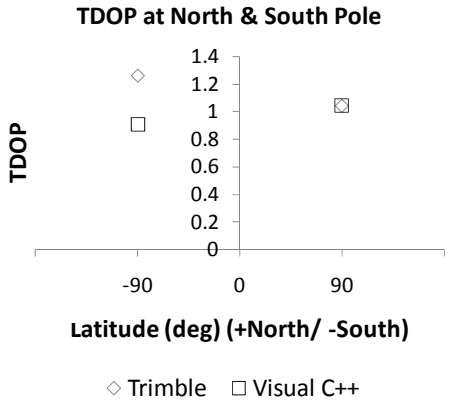



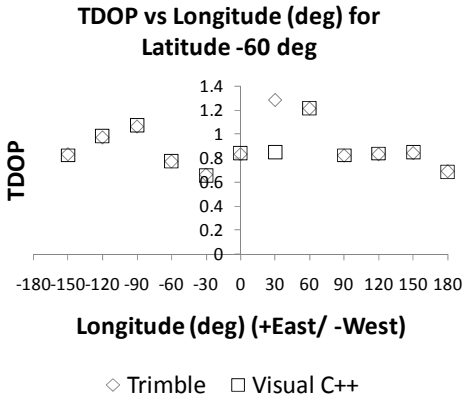



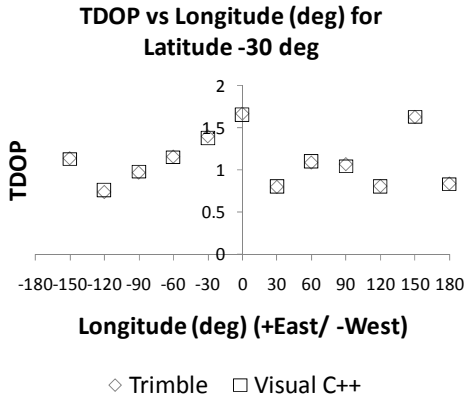



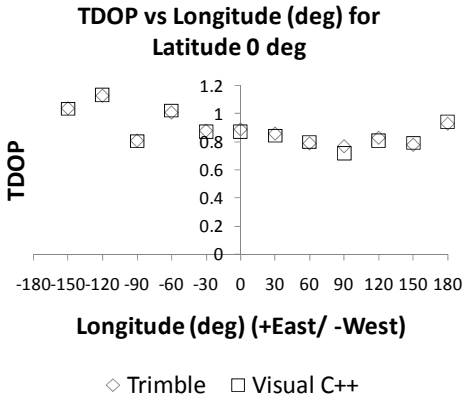



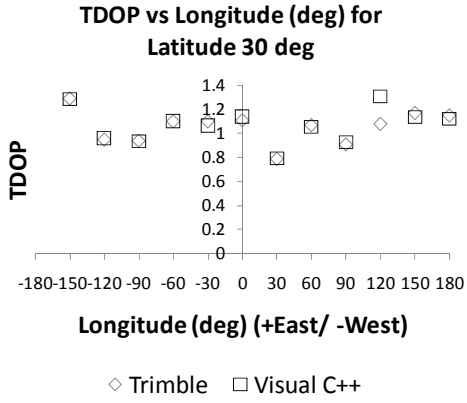
## B. TDOP

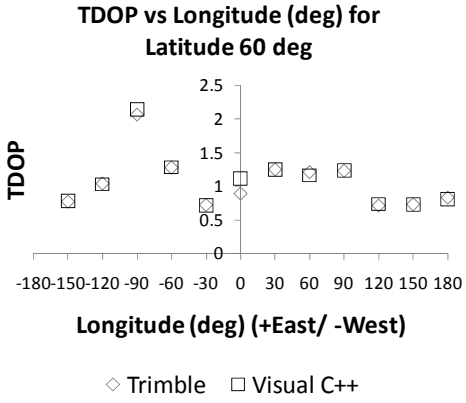

TDOP at North & South Pole
TDOP
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-90
0
90
Latitude (deg) (+North/ -South)
◇ Trimble □ Visual C++


TDOP vs Longitude (deg) for Latitude -60 deg
TDOP
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Visual C++

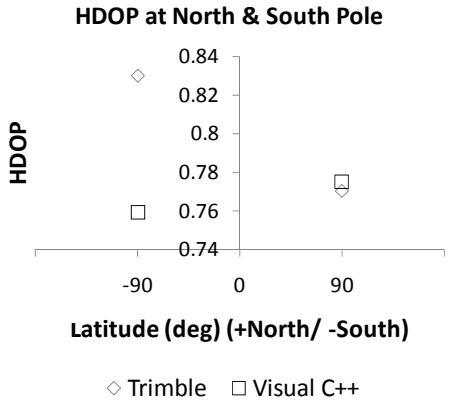

TDOP vs Longitude (deg) for Latitude -30 deg
TDOP
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Visual C++

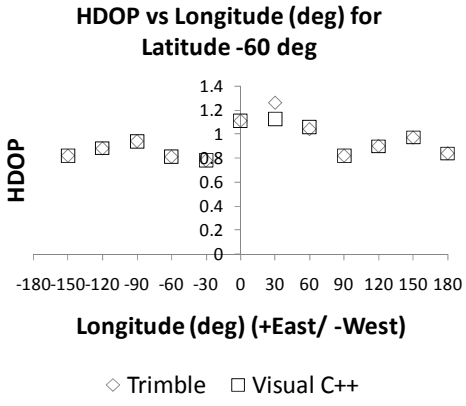

TDOP vs Longitude (deg) for Latitude 0 deg
TDOP
1.2
1
0.8
0.6
0.4
0.2
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Visual C++

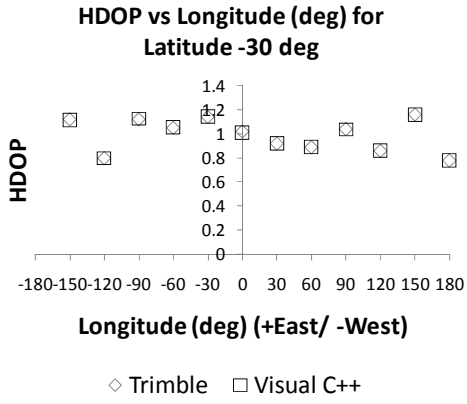

TDOP vs Longitude (deg) for Latitude 30 deg
TDOP
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Visual C++

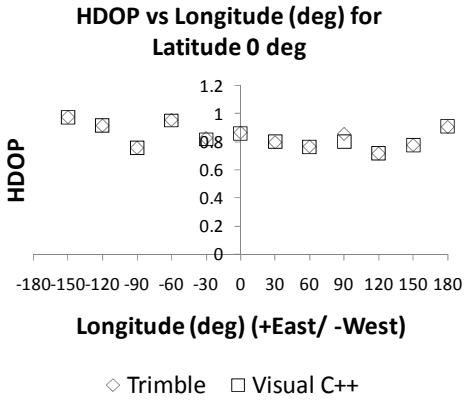

TDOP vs Longitude (deg) for Latitude 60 deg
TDOP
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Visual C++

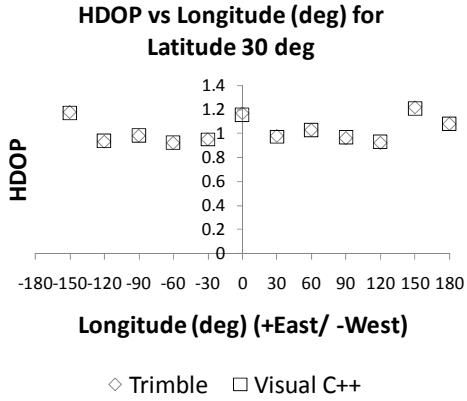
## C. HDOP

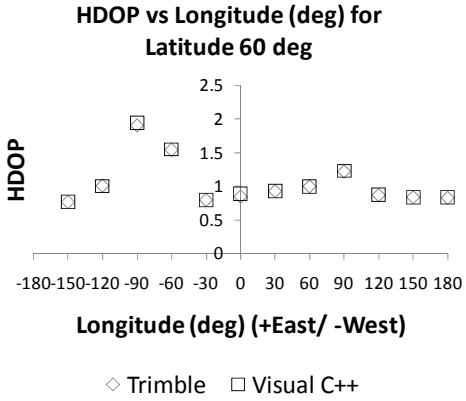

HDOP at North & South Pole
HDOP
0.84
0.82
0.8
0.78
0.76
0.74
-90
0
90
Latitude (deg) (+North/ -South)
◇ Trimble □ Visual C++


HDOP vs Longitude (deg) for Latitude -60 deg
HDOP
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Visual C++

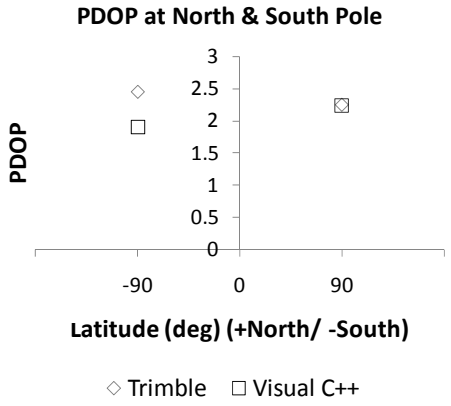

HDOP vs Longitude (deg) for Latitude -30 deg
HDOP
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Visual C++

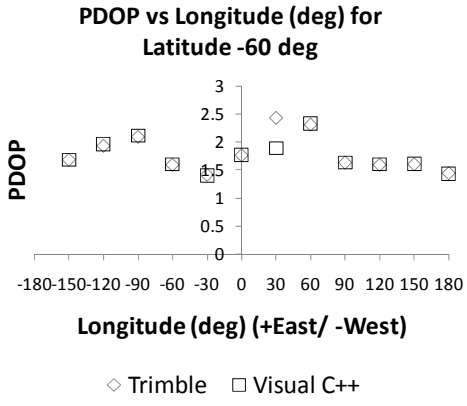

HDOP vs Longitude (deg) for Latitude 0 deg
HDOP
1.2
1
0.8
0.6
0.4
0.2
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Visual C++

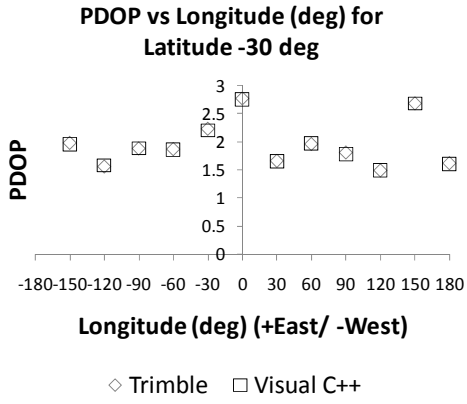

HDOP vs Longitude (deg) for Latitude 30 deg
HDOP
1.4
1.2
1
0.8
0.6
0.4
0.2
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Visual C++

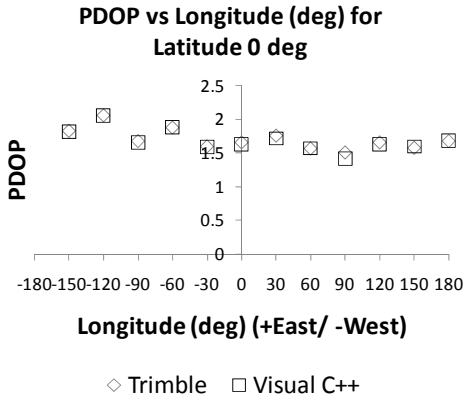

HDOP vs Longitude (deg) for Latitude 60 deg
HDOP
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Visual C++

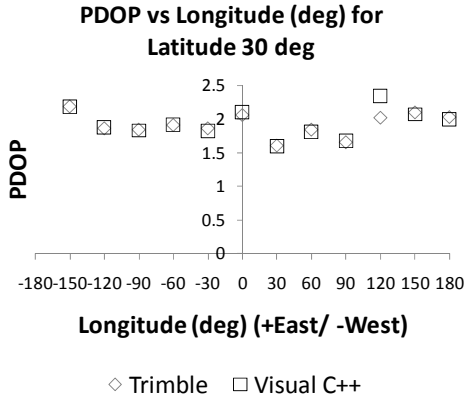
## D. PDOP

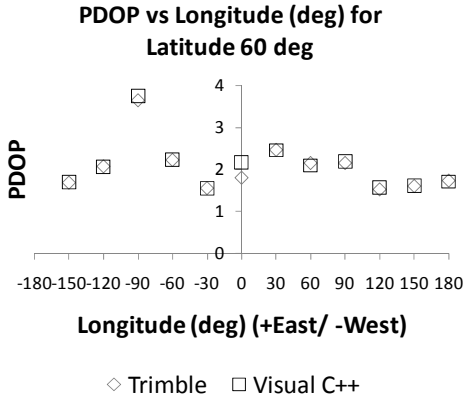

PDOP at North & South Pole
PDOP
3
2.5
2
1.5
1
0.5
0
-90
0
90
Latitude (deg) (+North/ -South)
◇ Trimble □ Visual C++


PDOP vs Longitude (deg) for Latitude -60 deg
PDOP
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Visual C++

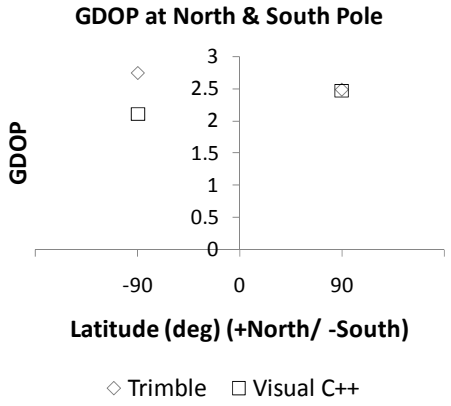

PDOP vs Longitude (deg) for Latitude -30 deg
PDOP
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Visual C++

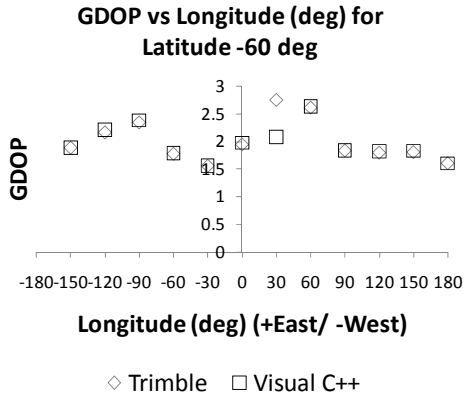

PDOP vs Longitude (deg) for Latitude 0 deg
PDOP
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Visual C++

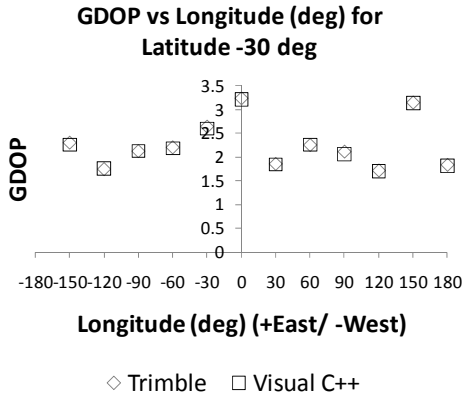

PDOP vs Longitude (deg) for Latitude 30 deg
PDOP
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Visual C++

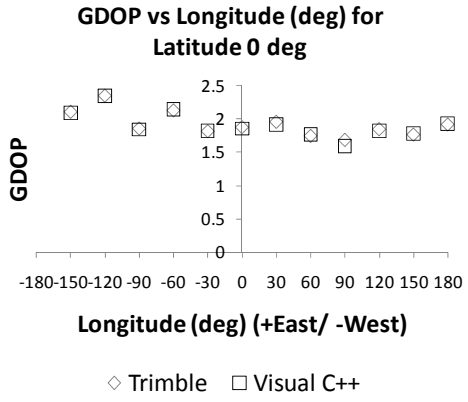

PDOP vs Longitude (deg) for Latitude 60 deg
PDOP
4
3
2
1
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Visual C++

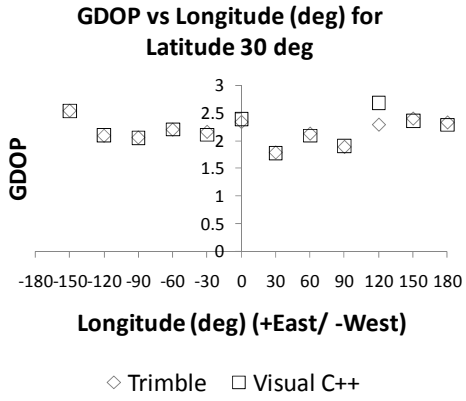
## E. GDOP

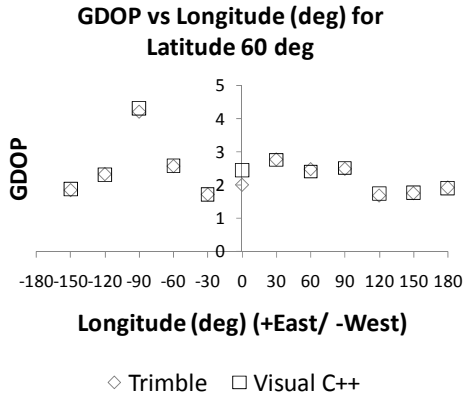

GDOP at North & South Pole
GDOP
3
2.5
2
1.5
1
0.5
0
-90
0
90
Latitude (deg) (+North/ -South)
◇ Trimble □ Visual C++


GDOP vs Longitude (deg) for Latitude -60 deg
GDOP
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Visual C++


GDOP vs Longitude (deg) for Latitude -30 deg
GDOP
3.5
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Visual C++


GDOP vs Longitude (deg) for Latitude 0 deg
GDOP
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Visual C++


GDOP vs Longitude (deg) for Latitude 30 deg
GDOP
3
2.5
2
1.5
1
0.5
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Visual C++


GDOP vs Longitude (deg) for Latitude 60 deg
GDOP
5
4
3
2
1
0
-180 -150 -120 -90 -60 -30 0 30 60 90 120 150 180
Longitude (deg) (+East/ -West)
◇ Trimble □ Visual C++

## F. DOP COMPARISON IN TABLE

| Location | | Trimble | | | | | | VC++ | | | | | | Percentage Difference | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Lat (+/-90) +North/-South) | Long (+/-180) (+East/-West) | GDOP | PDOP | HDOP | TDOP | VDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | XDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | Difference in Visible SV |
| -60 | -150 | 1.88 | 1.68 | 0.82 | 0.83 | 1.47 | 9 | 1.874 | 1.684 | 0.818 | 0.822 | 1.471 | 0.568 | 9 | -0.32 | 0.24 | -0.24 | -0.96 | 0.07 | 0 |
| | -120 | 2.16 | 1.93 | 0.88 | 0.97 | 1.72 | 9 | 2.195 | 1.961 | 0.881 | 0.985 | 1.753 | 0.514 | 9 | 1.62 | 1.61 | 0.11 | 1.55 | 1.92 | 0 |
| | -90 | 2.34 | 2.09 | 0.94 | 1.06 | 1.87 | 9 | 2.368 | 2.111 | 0.936 | 1.074 | 1.893 | 0.592 | 9 | 1.20 | 1.00 | -0.43 | 1.32 | 1.23 | 0 |
| | -60 | 1.77 | 1.59 | 0.81 | 0.77 | 1.37 | 10 | 1.78 | 1.604 | 0.81 | 0.772 | 1.384 | 0.556 | 10 | 0.56 | 0.88 | 0.00 | 0.26 | 1.02 | 0 |
| | -30 | 1.55 | 1.41 | 0.78 | 0.66 | 1.17 | 11 | 1.548 | 1.404 | 0.778 | 0.653 | 1.168 | 0.503 | 11 | -0.13 | -0.43 | -0.26 | -1.06 | -0.17 | 0 |
| | 0 | 1.94 | 1.76 | 1.11 | 0.83 | 1.36 | 8 | 1.965 | 1.775 | 1.106 | 0.843 | 1.388 | 0.922 | 8 | 1.29 | 0.85 | -0.36 | 1.57 | 2.06 | 0 |
| | 30 | 2.75 | 2.43 | 1.26 | 1.28 | 2.08 | 8 | 2.07 | 1.888 | 1.124 | 0.849 | 1.517 | 1.032 | 9 | 24.73 | 22.30 | 10.79 | 33.67 | 27.07 | -1 |
| | 60 | 2.61 | 2.31 | 1.04 | 1.21 | 2.06 | 9 | 2.624 | 2.326 | 1.056 | 1.216 | 2.072 | 0.912 | 9 | 0.54 | 0.69 | 1.54 | 0.50 | 0.58 | 0 |
| | 90 | 1.83 | 1.63 | 0.82 | 0.82 | 1.41 | 10 | 1.83 | 1.634 | 0.818 | 0.824 | 1.415 | 0.56 | 10 | 0.00 | 0.25 | -0.24 | 0.49 | 0.35 | 0 |
| | 120 | 1.79 | 1.59 | 0.9 | 0.83 | 1.31 | 9 | 1.809 | 1.604 | 0.896 | 0.837 | 1.331 | 0.638 | 9 | 1.06 | 0.88 | -0.44 | 0.84 | 1.60 | 0 |
| | 150 | 1.8 | 1.6 | 0.97 | 0.84 | 1.27 | 9 | 1.821 | 1.611 | 0.971 | 0.849 | 1.285 | 0.673 | 9 | 1.17 | 0.69 | 0.10 | 1.07 | 1.18 | 0 |
| | 180 | 1.6 | 1.44 | 0.84 | 0.69 | 1.17 | 10 | 1.594 | 1.44 | 0.835 | 0.684 | 1.173 | 0.664 | 10 | -0.38 | 0.00 | -0.60 | -0.87 | 0.26 | 0 |
| -30 | -150 | 2.3 | 1.99 | 1.12 | 1.14 | 1.64 | 9 | 2.263 | 1.963 | 1.114 | 1.127 | 1.616 | 0.612 | 9 | -1.61 | -1.36 | -0.54 | -1.14 | -1.46 | 0 |
| | -120 | 1.74 | 1.57 | 0.8 | 0.74 | 1.35 | 10 | 1.763 | 1.592 | 0.801 | 0.756 | 1.376 | 0.615 | 10 | 1.32 | 1.40 | 0.13 | 2.16 | 1.93 | 0 |
| | -90 | 2.12 | 1.89 | 1.12 | 0.97 | 1.51 | 8 | 2.13 | 1.893 | 1.126 | 0.977 | 1.522 | 0.978 | 8 | 0.47 | 0.16 | 0.54 | 0.72 | 0.79 | 0 |
| | -60 | 2.2 | 1.88 | 1.05 | 1.16 | 1.55 | 9 | 2.191 | 1.865 | 1.057 | 1.15 | 1.536 | 0.783 | 9 | -0.41 | -0.80 | 0.67 | -0.86 | -0.90 | 0 |
| | -30 | 2.64 | 2.24 | 1.14 | 1.4 | 1.92 | 8 | 2.607 | 2.214 | 1.143 | 1.377 | 1.896 | 0.825 | 8 | -1.25 | -1.16 | 0.26 | -1.64 | -1.25 | 0 |
| | 0 | 3.24 | 2.78 | 1.02 | 1.67 | 2.58 | 9 | 3.227 | 2.767 | 1.011 | 1.659 | 2.576 | 0.769 | 9 | -0.40 | -0.47 | -0.88 | -0.66 | -0.16 | 0 |
| | 30 | 1.85 | 1.67 | 0.92 | 0.8 | 1.4 | 9 | 1.844 | 1.66 | 0.923 | 0.802 | 1.381 | 0.643 | 9 | -0.32 | -0.60 | 0.33 | 0.25 | -1.36 | 0 |
| | 60 | 2.25 | 1.97 | 0.89 | 1.09 | 1.76 | 9 | 2.265 | 1.979 | 0.893 | 1.102 | 1.767 | 0.659 | 9 | 0.67 | 0.46 | 0.34 | 1.10 | 0.40 | 0 |
| | 90 | 2.11 | 1.82 | 1.04 | 1.07 | 1.49 | 9 | 2.068 | 1.786 | 1.038 | 1.042 | 1.454 | 0.706 | 9 | -1.99 | -1.87 | -0.19 | -2.62 | -2.42 | 0 |
| | 120 | 1.71 | 1.5 | 0.86 | 0.81 | 1.23 | 10 | 1.698 | 1.498 | 0.862 | 0.8 | 1.225 | 0.658 | 10 | -0.70 | -0.13 | 0.23 | -1.23 | -0.41 | 0 |
| | 150 | 3.15 | 2.69 | 1.16 | 1.63 | 2.43 | 9 | 3.145 | 2.691 | 1.161 | 1.628 | 2.427 | 0.775 | 9 | -0.16 | 0.04 | 0.09 | -0.12 | -0.12 | 0 |
| | 180 | 1.83 | 1.63 | 0.78 | 0.84 | 1.43 | 11 | 1.813 | 1.614 | 0.78 | 0.826 | 1.413 | 0.573 | 11 | -0.93 | -0.98 | 0.00 | -1.67 | -1.19 | 0 |

| Location | | Trimble | | | | | | VC++ | | | | | | | | Percentage Difference | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Lat (+/-90) (+North/-South) | Long (+/-180) (+East/-West) | GDOP | PDOP | HDOP | TDOP | VDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | Difference in Visible SV |
| 0 | -150 | 2.11 | 1.83 | 0.98 | 1.04 | 1.54 | 9 | 2.092 | 1.817 | 0.978 | 1.037 | 1.532 | 0.684 | 0.7 | 9 | -0.85 | -0.71 | -0.20 | -0.29 | -0.52 | 0 |
| | -120 | 2.35 | 2.06 | 0.92 | 1.13 | 1.85 | 8 | 2.355 | 2.062 | 0.92 | 1.137 | 1.845 | 0.63 | 0.671 | 8 | 0.21 | 0.10 | 0.00 | 0.62 | -0.27 | 0 |
| | -90 | 1.86 | 1.68 | 0.76 | 0.81 | 1.49 | 11 | 1.846 | 1.662 | 0.758 | 0.804 | 1.479 | 0.542 | 0.53 | 11 | -0.75 | -1.07 | -0.26 | -0.74 | -0.74 | 0 |
| | -60 | 2.13 | 1.88 | 0.96 | 1.01 | 1.62 | 10 | 2.146 | 1.887 | 0.954 | 1.022 | 1.628 | 0.519 | 0.801 | 10 | 0.75 | 0.37 | -0.63 | 1.19 | 0.49 | 0 |
| | -30 | 1.82 | 1.6 | 0.83 | 0.88 | 1.37 | 11 | 1.822 | 1.6 | 0.82 | 0.871 | 1.374 | 0.529 | 0.626 | 11 | 0.11 | 0.00 | -1.20 | -1.02 | 0.29 | 0 |
| | 0 | 1.88 | 1.66 | 0.87 | 0.89 | 1.41 | 10 | 1.858 | 1.64 | 0.863 | 0.873 | 1.395 | 0.559 | 0.657 | 10 | -1.17 | -1.20 | -0.80 | -1.91 | -1.06 | 0 |
| | 30 | 1.96 | 1.76 | 0.8 | 0.86 | 1.56 | 10 | 1.919 | 1.723 | 0.806 | 0.844 | 1.523 | 0.649 | 0.477 | 10 | -2.09 | -2.10 | 0.75 | -1.86 | -2.37 | 0 |
| | 60 | 1.75 | 1.57 | 0.77 | 0.79 | 1.37 | 11 | 1.769 | 1.578 | 0.766 | 0.799 | 1.38 | 0.534 | 0.549 | 11 | 1.09 | 0.51 | -0.52 | 1.14 | 0.73 | 0 |
| | 90 | 1.69 | 1.51 | 0.86 | 0.77 | 1.24 | 11 | 1.592 | 1.422 | 0.806 | 0.717 | 1.171 | 0.531 | 0.607 | 11 | -5.80 | -5.83 | -6.28 | -6.88 | -5.56 | 0 |
| | 120 | 1.85 | 1.66 | 0.72 | 0.83 | 1.49 | 12 | 1.828 | 1.64 | 0.72 | 0.808 | 1.473 | 0.476 | 0.54 | 12 | -1.19 | -1.20 | 0.00 | -2.65 | -1.14 | 0 |
| | 150 | 1.77 | 1.58 | 0.78 | 0.78 | 1.38 | 11 | 1.785 | 1.601 | 0.78 | 0.791 | 1.398 | 0.58 | 0.521 | 11 | 0.85 | 1.33 | 0.00 | 1.41 | 1.30 | 0 |
| | 180 | 1.92 | 1.68 | 0.91 | 0.93 | 1.41 | 9 | 1.935 | 1.691 | 0.912 | 0.942 | 1.424 | 0.579 | 0.705 | 9 | 0.78 | 0.65 | 0.22 | 1.29 | 0.99 | 0 |
| 30 | -150 | 2.54 | 2.19 | 1.18 | 1.29 | 1.84 | 8 | 2.531 | 2.179 | 1.169 | 1.288 | 1.838 | 0.865 | 0.786 | 8 | -0.35 | -0.50 | -0.93 | -0.16 | -0.11 | 0 |
| | -120 | 2.09 | 1.86 | 0.93 | 0.95 | 1.61 | 9 | 2.105 | 1.873 | 0.938 | 0.96 | 1.621 | 0.675 | 0.652 | 9 | 0.72 | 0.70 | 0.86 | 1.05 | 0.68 | 0 |
| | -90 | 2.06 | 1.84 | 0.98 | 0.94 | 1.55 | 10 | 2.056 | 1.831 | 0.982 | 0.936 | 1.545 | 0.82 | 0.541 | 10 | -0.19 | -0.49 | 0.20 | -0.43 | -0.32 | 0 |
| | -60 | 2.2 | 1.91 | 0.92 | 1.1 | 1.67 | 9 | 2.208 | 1.914 | 0.921 | 1.102 | 1.677 | 0.672 | 0.63 | 9 | 0.36 | 0.21 | 0.11 | 0.18 | 0.42 | 0 |
| | -30 | 2.16 | 1.86 | 0.95 | 1.1 | 1.59 | 9 | 2.113 | 1.825 | 0.948 | 1.065 | 1.56 | 0.714 | 0.623 | 9 | -2.18 | -1.88 | -0.21 | -3.18 | -1.89 | 0 |
| | 0 | 2.34 | 2.06 | 1.17 | 1.11 | 1.69 | 8 | 2.391 | 2.1 | 1.152 | 1.143 | 1.756 | 1.006 | 0.561 | 8 | 2.18 | 1.94 | -1.54 | 2.97 | 3.91 | 0 |
| | 30 | 1.79 | 1.6 | 0.98 | 0.79 | 1.27 | 9 | 1.782 | 1.596 | 0.972 | 0.793 | 1.266 | 0.821 | 0.522 | 9 | -0.45 | -0.25 | -0.82 | 0.38 | -0.31 | 0 |
| | 60 | 2.13 | 1.84 | 1.03 | 1.07 | 1.53 | 8 | 2.097 | 1.813 | 1.029 | 1.054 | 1.493 | 0.771 | 0.681 | 8 | -1.55 | -1.47 | -0.10 | -1.50 | -2.42 | 0 |
| | 90 | 1.89 | 1.65 | 0.96 | 0.91 | 1.34 | 9 | 1.915 | 1.675 | 0.966 | 0.928 | 1.368 | 0.743 | 0.618 | 9 | 1.32 | 1.52 | 0.63 | 1.98 | 2.09 | 0 |
| | 120 | 2.29 | 2.02 | 0.92 | 1.08 | 1.8 | 10 | 2.683 | 2.341 | 0.931 | 1.309 | 2.148 | 0.766 | 0.529 | 9 | 17.16 | 15.89 | 1.20 | 21.20 | 19.33 | 1 |
| | 150 | 2.4 | 2.1 | 1.22 | 1.17 | 1.71 | 7 | 2.361 | 2.068 | 1.208 | 1.138 | 1.679 | 1.005 | 0.671 | 7 | -1.62 | -1.52 | -0.98 | -2.74 | -1.81 | 0 |
| | 180 | 2.33 | 2.03 | 1.08 | 1.15 | 1.71 | 8 | 2.287 | 1.993 | 1.08 | 1.122 | 1.675 | 0.851 | 0.665 | 8 | -1.85 | -1.82 | 0.00 | -2.43 | -2.05 | 0 |

| Location | | Trimble | | | | | | VC++ | | | | | | | | Percentage Difference | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Lat (+/-90) (+North/-South) | Long (+/-180) (+East/-West) | GDOP | PDOP | HDOP | TDOP | VDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | Difference in Visible SV |
| 60 | -150 | 1.85 | 1.69 | 0.77 | 0.77 | 1.5 | 10 | 1.872 | 1.699 | 0.767 | 0.786 | 1.515 | 0.544 | 0.541 | 10 | 1.19 | 0.53 | -0.39 | 2.08 | 1.00 | 0 |
| | -120 | 2.33 | 2.08 | 1.01 | 1.04 | 1.82 | 8 | 2.303 | 2.058 | 1.003 | 1.034 | 1.797 | 0.756 | 0.659 | 8 | -1.16 | -1.06 | -0.69 | -0.58 | -1.26 | 0 |
| | -90 | 4.2 | 3.65 | 1.9 | 2.07 | 3.11 | 6 | 4.312 | 3.742 | 1.946 | 2.143 | 3.196 | 1.683 | 0.976 | 6 | 2.67 | 2.52 | 2.42 | 3.53 | 2.77 | 0 |
| | -60 | 2.57 | 2.23 | 1.54 | 1.28 | 1.61 | 7 | 2.578 | 2.233 | 1.551 | 1.287 | 1.607 | 1.362 | 0.741 | 7 | 0.31 | 0.13 | 0.71 | 0.55 | -0.19 | 0 |
| | -30 | 1.72 | 1.56 | 0.8 | 0.72 | 1.34 | 10 | 1.711 | 1.553 | 0.799 | 0.719 | 1.332 | 0.586 | 0.544 | 10 | -0.52 | -0.45 | -0.13 | -0.14 | -0.60 | 0 |
| | 0 | 2.01 | 1.81 | 0.85 | 0.89 | 1.6 | 10 | 2.437 | 2.167 | 0.892 | 1.117 | 1.974 | 0.721 | 0.526 | 9 | 21.24 | 19.72 | 4.94 | 25.51 | 23.38 | 1 |
| | 30 | 2.77 | 2.47 | 0.93 | 1.25 | 2.29 | 9 | 2.758 | 2.459 | 0.927 | 1.248 | 2.278 | 0.746 | 0.55 | 9 | -0.43 | -0.45 | -0.32 | -0.16 | -0.52 | 0 |
| | 60 | 2.48 | 2.16 | 1 | 1.21 | 1.92 | 8 | 2.402 | 2.1 | 1 | 1.167 | 1.846 | 0.783 | 0.622 | 8 | -3.15 | -2.78 | 0.00 | -3.55 | -3.85 | 0 |
| | 90 | 2.49 | 2.16 | 1.22 | 1.23 | 1.79 | 8 | 2.515 | 2.188 | 1.229 | 1.24 | 1.81 | 0.775 | 0.953 | 7 | 1.00 | 1.30 | 0.74 | 0.81 | 1.12 | 1 |
| | 120 | 1.69 | 1.53 | 0.87 | 0.71 | 1.26 | 9 | 1.74 | 1.574 | 0.88 | 0.742 | 1.305 | 0.688 | 0.548 | 9 | 2.96 | 2.88 | 1.15 | 4.51 | 3.57 | 0 |
| | 150 | 1.76 | 1.61 | 0.84 | 0.73 | 1.37 | 9 | 1.777 | 1.618 | 0.837 | 0.733 | 1.385 | 0.668 | 0.505 | 9 | 0.97 | 0.50 | -0.36 | 0.41 | 1.09 | 0 |
| | 180 | 1.93 | 1.74 | 0.83 | 0.83 | 1.53 | 9 | 1.91 | 1.716 | 0.835 | 0.816 | 1.5 | 0.561 | 0.618 | 9 | -1.04 | -1.38 | 0.60 | -1.69 | -1.96 | 0 |
| 90 | 0 | 2.48 | 2.24 | 0.77 | 1.04 | 2.11 | 9 | 2.465 | 2.233 | 0.775 | 1.043 | 2.095 | 0.557 | 0.539 | 9 | -0.60 | -0.31 | 0.65 | 0.29 | -0.71 | 0 |
| -90 | 0 | 2.74 | 2.44 | 0.83 | 1.26 | 2.3 | 9 | 2.106 | 1.901 | 0.759 | 0.906 | 1.743 | 0.559 | 0.514 | 10 | -23.14 | -22.09 | -8.55 | -28.10 | -24.22 | -1 |

# APPENDIX F. COMPARISON OF VISIBLE SATELLITES BETWEEN MATLAB & TRIMBLE

| Location | Lat (+/-90) (+North/-South) | Angle between GPS Receiver & Satellite (degrees) wrt Vertical | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Long (+/-180) (+East/-West) | -75 | | -75 | | -60 | | -60 | | -60 | | -60 | | -60 | | -60 | | -45 | | -45 | | -45 | | -45 | |
| | | -120 | | -45 | | -105 | | -75 | | -45 | | 30 | | 75 | | 105 | | -150 | | -105 | | | | | |
| Program | | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble |
| Satellite Number | 2 | 74.54 | 73.83 | 60.61 | 59.90 | | | 79.53 | 78.86 | 66.57 | 65.88 | 24.77 | 23.64 | 31.08 | 30.70 | 48.87 | 48.73 | | | | | | | | |
| | 3 | | | | | | | | | | | | | | | | | 59.26 | 59.65 | 79.81 | | | | | |
| | 4 | | | | | | | | | | | 63.08 | 63.41 | 50.19 | 51.00 | 56.53 | 56.68 | | | | | | | | |
| | 5 | | | | | | | 79.84 | | 72.57 | 72.89 | | | | | | | | | | | | | | |
| | 6 | | | | | 78.38 | 77.69 | | | | | | | | | | | 51.22 | 50.80 | | | 68.75 | 67.70 | | |
| | 7 | | | | | | | | | | | 77.18 | 76.08 | 53.99 | 53.54 | 46.19 | 45.60 | | | | | | | | |
| | 8 | | | | | | | | | | | | | 79.91 | | 79.88 | | | | | | | | | |
| | 9 | | | | | | | | | | | | | 60.01 | 59.74 | 75.71 | 75.76 | | | | | | | | |
| | 10 | 74.68 | 74.87 | 54.47 | 53.39 | 80.04 | 79.87 | 65.42 | 64.90 | 48.19 | 47.22 | 34.77 | 34.69 | | | | | | | | | | | | |
| | 11 | | | | | | | | | | | | | | | | | | | | | | | | |
| | 12 | | | | | | | | | 80.38 | 79.80 | | | | | | | | | | | | | | |
| | 13 | 52.21 | 51.77 | 59.82 | 59.83 | 69.51 | 69.79 | 75.40 | 74.81 | 76.31 | 75.83 | 57.16 | 56.87 | 33.42 | 32.89 | 14.36 | 13.16 | 63.51 | 63.78 | | | | | | |
| | 14 | | | | | | | | | | | | | | | | | | | | | | | | |
| | 15 | | | | | | | | | | | | | | | | | | | | | | | | |
| | 16 | 39.30 | 38.69 | 60.90 | 60.77 | 40.60 | 39.73 | 57.87 | 56.76 | 71.69 | 70.77 | | | 77.08 | 76.86 | 65.44 | 64.88 | 8.24 | 7.77 | | | 45.17 | 44.76 | | |
| | 17 | | | | | | | | | | | | | | | | | | | | | | | | |
| | 18 | | | | | | | | | | | | | | | | | | | | | | | | |
| | 19 | | | | | | | | | | | | | | | | | | | | | | | | |
| | 20 | | | | | | | | | | | | | | | | | | | | | | | | |
| | 21 | 79.62 | 78.55 | 78.25 | 77.68 | 63.08 | 62.66 | 58.77 | 58.65 | 65.65 | 64.68 | | | | | | | | | | | 49.43 | 48.56 | | |
| | 22 | | | | | | | | | | | | | | | | | | | | | | | | |
| | 23 | 52.71 | 52.73 | 69.80 | 69.80 | 63.84 | 63.76 | 76.58 | 76.78 | | | | | 62.76 | 61.88 | 46.06 | 44.04 | 40.71 | 40.75 | | | 72.95 | 72.77 | | |
| | 24 | 58.17 | 57.16 | 44.66 | 43.65 | 49.50 | 48.14 | 32.60 | 31.49 | 26.13 | 25.65 | 63.95 | 62.75 | | | | | | | | | 50.76 | 50.90 | | |
| | 25 | 74.94 | 73.77 | 80.53 | 79.84 | | | | | | | 69.69 | 69.89 | 44.11 | 43.29 | 30.44 | 29.74 | 80.17 | 78.76 | | | | | | |
| | 26 | | | | | | | | | | | | | | | | | | | | | | | | |
| | 27 | | | | | | | | | | | | | | | | | | | | | | | | |
| | 28 | | | | | | | | | | | | | | | | | | | | | | | | |
| | 29 | 42.04 | 41.08 | 26.41 | 25.65 | 37.49 | 36.90 | 18.52 | 17.20 | 7.71 | 7.67 | 51.93 | 51.76 | 70.55 | 70.79 | 77.31 | 76.81 | 72.82 | 72.86 | | | 45.53 | 44.87 | | |
| | 30 | 79.37 | | 67.26 | 67.64 | 69.22 | 69.28 | 55.35 | 56.61 | 50.04 | 50.64 | 79.85 | | | | | | | | | | 65.46 | 66.18 | | |
| | 31 | 62.77 | 61.87 | 72.51 | 71.71 | 44.84 | 43.65 | 53.02 | 52.69 | 68.37 | 67.72 | | | | | | | 49.51 | 48.27 | | | 26.27 | 25.65 | | |
| | 32 | | | | | | | | | | | | | | | | | | | | | | | | |

| | | Angle between GPS Receiver & Satellite (degrees) wrt Vertical | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Location | Lat (+/-90) (+North/-South) | -45 | | -45 | | -45 | | -30 | | -30 | | -15 | | -15 | | -15 | | -15 | | 0 | |
| | Long (+/-180) (+East/-West) | -75 | | 60 | | 135 | | -75 | | 75 | | -90 | | 30 | | 105 | | 120 | | -165 | |
| Program | | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble |
| Satellite Number | 2 | | | 13.31 | 13.73 | 73.66 | 73.76 | | | 34.54 | 35.77 | | | 40.52 | 40.84 | 70.45 | 70.77 | | | | |
| | 3 | | | | | | | | | | | | | | | | | | | 9.87 | 8.36 |
| | 4 | | | 35.49 | 35.68 | 67.68 | 67.72 | | | 12.53 | 12.90 | | | 54.35 | 54.90 | 37.59 | 36.74 | 54.51 | 53.74 | | |
| | 5 | 68.75 | 69.50 | | | | | 58.95 | 59.41 | | | 68.10 | 68.20 | 80.23 | 79.72 | | | | | | |
| | 6 | | | | | | | | | | | 72.28 | 71.73 | | | | | | | 21.51 | 21.89 |
| | 7 | | | 55.44 | 54.23 | 39.31 | 39.70 | | | 38.47 | 37.15 | | | | | 12.04 | 11.84 | 18.88 | 18.77 | | |
| | 8 | | | 69.96 | 69.64 | 77.82 | 77.69 | | | 48.34 | 47.76 | | | 76.47 | 75.24 | 32.44 | 32.68 | 45.85 | 45.70 | | |
| | 9 | | | | | | | | | | | | | | | | | | | | |
| | 10 | 70.78 | 70.89 | 51.87 | 51.74 | | | 78.19 | 77.88 | 70.36 | 69.76 | | | 41.08 | 40.78 | | | | | | |
| | 11 | | | | | | | | | | | | | | | | | 80.03 | 78.86 | 79.50 | |
| | 12 | 80.09 | 79.45 | | | | | 71.72 | 70.38 | | | 80.22 | 79.22 | 72.22 | 71.72 | | | | | | |
| | 13 | | | 52.07 | 51.87 | 17.80 | 17.80 | | | 55.40 | 54.86 | | | | | 56.33 | 55.84 | 53.37 | 52.82 | | |
| | 14 | | | | | | | | | | | 70.27 | 69.67 | | | | | | | 65.23 | 64.54 |
| | 15 | | | | | | | | | | | | | 56.56 | 56.67 | | | | | | |
| | 16 | 66.46 | 65.77 | | | 56.57 | 56.88 | 76.78 | 75.77 | | | 77.49 | 76.78 | | | | | | | 61.16 | 60.83 |
| | 17 | | | | | | | | | | | | | 74.59 | 74.70 | 78.84 | 78.68 | | | | |
| | 18 | | | | | | | 80.48 | 79.85 | | | 71.39 | 70.69 | | | | | | | | |
| | 19 | | | | | | | | | | | | | | | | | | | 40.06 | 39.65 |
| | 20 | | | | | 56.97 | 56.88 | | | | | | | | | 56.46 | 56.24 | 40.07 | 39.34 | 58.61 | 58.73 |
| | 21 | 41.01 | 40.65 | | | | | 22.35 | 21.65 | | | 14.65 | 14.17 | | | | | | | | |
| | 22 | | | | | | | | | | | 79.43 | 78.64 | | | | | | | | |
| | 23 | | | 80.28 | 79.87 | 26.20 | 24.89 | | | | | | | | | 71.12 | 70.86 | 60.35 | 59.86 | 66.74 | 66.79 |
| | 24 | 24.67 | 24.14 | | | | | 29.50 | 29.88 | | | 54.07 | 54.87 | | | | | | | | |
| | 25 | | | 53.67 | 53.90 | 20.10 | 18.71 | | | 45.24 | 45.89 | | | | | 32.53 | 33.84 | 29.83 | 30.82 | | |
| | 26 | | | 79.21 | | | | | | 77.81 | | | | 36.07 | 38.65 | | | | | | |
| | 27 | | | | | | | | | 66.09 | 63.65 | | | 67.19 | 66.52 | 63.99 | 61.68 | 75.75 | 73.69 | | |
| | 28 | | | | | | | | | | | | | | | 79.04 | 77.65 | | | | |
| | 29 | 24.27 | 23.87 | 77.96 | 77.78 | | | 39.28 | 38.85 | | | 62.57 | 61.85 | | | | | | | | |
| | 30 | 42.64 | 43.48 | | | | | 34.25 | 34.26 | | | 51.52 | 51.90 | | | | | | | | |
| | 31 | 43.00 | 42.71 | | | | | 38.66 | 38.73 | | | 26.02 | 26.76 | | | | | | | 73.40 | 72.88 |
| | 32 | | | | | 80.32 | 78.68 | | | | | | | | | 81.38 | 79.33 | 67.65 | 65.41 | 41.62 | 41.70 |

| | | Angle between GPS Receiver & Satellite (degrees) wrt Vertical | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Location | Lat (+/-90) (+North/-South) | 0 | | 0 | | 0 | | 0 | | 15 | | 15 | | 15 | | 15 | | 30 | | 45 | |
| | Long (+/-180) (+East/-West) | -45 | | 45 | | 90 | | 135 | | -90 | | -45 | | 0 | | 75 | | 120 | | -45 | |
| Program | | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble |
| Satellite Number | 2 | | | 54.65 | 54.82 | 71.61 | 71.78 | | | | | | | | | 77.56 | 77.80 | | | | |
| | 3 | | | | | | | | | | | | | | | | | | | | |
| | 4 | | | 45.55 | 45.87 | 32.56 | 31.79 | 75.63 | 74.75 | | | | | | | 45.04 | 43.82 | 80.04 | 78.77 | | |
| | 5 | 12.02 | 12.09 | | | | | | | 68.74 | 68.90 | 23.78 | 22.85 | 49.41 | 48.75 | | | | | 58.24 | 56.83 |
| | 6 | | | | | | | | | 75.06 | 74.74 | | | | | | | | | | |
| | 7 | | | 78.95 | 78.88 | 37.81 | 36.86 | 45.25 | 44.77 | | | | | | | 62.51 | 61.86 | 67.60 | 66.81 | | |
| | 8 | | | 57.07 | 56.18 | 6.86 | 6.64 | 56.89 | 56.72 | | | | | | | 23.20 | 22.88 | 47.80 | 47.77 | | |
| | 9 | 65.68 | 65.90 | | | | | | | 71.89 | 71.62 | 48.76 | 48.87 | 56.86 | 56.67 | | | | | 14.24 | 14.54 |
| | 10 | 72.63 | 72.86 | 65.51 | 65.78 | | | | | | | | | 65.69 | 65.83 | | | | | | |
| | 11 | | | | | 80.08 | 78.65 | 60.78 | 59.40 | | | | | | | 78.00 | 76.56 | 32.90 | 31.63 | | |
| | 12 | 25.65 | 24.25 | | | | | | | 77.16 | 76.08 | 27.46 | 26.88 | 35.35 | 35.75 | | | | | 54.25 | 53.85 |
| | 13 | | | | | 77.12 | 76.84 | 70.58 | 70.81 | | | | | | | | | | | | |
| | 14 | | | | | | | | | 43.81 | 43.69 | | | | | | | | | 71.32 | 71.71 |
| | 15 | 71.78 | 71.41 | 53.70 | 53.69 | | | | | | | 64.61 | 63.31 | 18.35 | 17.62 | 74.98 | 74.71 | | | 59.57 | 58.05 |
| | 16 | | | | | | | | | | | | | | | | | | | | |
| | 17 | | | 52.21 | 51.77 | 56.19 | 55.67 | | | | | | | 73.48 | 72.45 | 32.08 | 31.66 | 58.58 | 58.71 | | |
| | 18 | 47.58 | 46.66 | | | | | | | 44.32 | 44.50 | 30.68 | 28.67 | 68.92 | 67.70 | | | | | 18.57 | 18.77 |
| | 19 | | | | | | | 73.34 | 72.47 | | | | | | | | | 72.13 | 71.30 | | |
| | 20 | | | | | 71.41 | 70.09 | 17.39 | 16.07 | | | | | | | | | 51.80 | 51.87 | | |
| | 21 | 45.96 | 45.75 | | | | | | | 39.02 | 38.84 | 55.09 | 54.77 | | | | | | | 79.33 | 78.78 |
| | 22 | 77.68 | 77.67 | | | | | | | 45.92 | 45.64 | 64.05 | 63.68 | | | | | | | 39.92 | 39.70 |
| | 23 | | | | | | | 65.71 | 65.84 | | | | | | | | | | | | |
| | 24 | 51.04 | 50.82 | | | | | | | 79.95 | | 67.98 | 67.82 | | | | | | | | |
| | 25 | | | | | 56.72 | 57.85 | 52.22 | 52.80 | | | | | | | 79.36 | | | | | |
| | 26 | | | 30.67 | 32.70 | 79.41 | | | | | | 81.19 | 78.19 | 31.30 | 28.12 | 60.74 | 61.72 | | | 80.45 | 77.05 |
| | 27 | | | 42.81 | 42.53 | 39.36 | 36.68 | 81.12 | 79.71 | | | | | 79.26 | | 13.01 | 9.67 | 56.57 | 55.73 | | |
| | 28 | | | 79.74 | | 62.60 | 61.40 | 73.32 | 71.67 | | | | | | | 49.57 | 49.77 | 35.02 | 33.67 | | |
| | 29 | 67.81 | 67.82 | | | | | | | | | | | | | | | | | | |
| | 30 | 26.71 | 25.82 | | | | | | | 67.40 | 66.87 | 45.14 | 43.82 | 71.68 | 70.77 | | | | | 78.79 | 77.82 |
| | 31 | 79.79 | | | | | | | | 55.04 | 55.80 | | | | | | | | | | |
| | 32 | | | | | | | 44.07 | 41.37 | | | | | | | | | 56.09 | 55.90 | | |

| | | Angle between GPS Receiver & Satellite (degrees) wrt Vertical | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Location | Lat (+/-90) (+North/-South) | 45 | | 45 | | 45 | | 60 | | 60 | | 60 | | 75 | | 75 | | 75 | | -90 | |
| | Long (+/-180) (+East/-West) | -30 | | 60 | | 90 | | -45 | | 0 | | 90 | | -15 | | 75 | | 165 | | 0 | |
| Program | | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble | Matlab | Trimble |
| Satellite Number | 2 | | | | | | | | | | | | | | | | | | | 58.23 | 57.55 |
| | 3 | | | | | | | | | | | | | | | | | | | | |
| | 4 | | | 80.26 | 79.84 | 80.29 | 79.81 | | | | | | | | | | | | | | |
| | 5 | 57.77 | 56.81 | | | | | 74.56 | 73.83 | 80.57 | 79.78 | | | | | | | | | | |
| | 6 | | | | | | | | | | | | | | | | | | | | |
| | 7 | | | | | | | | | | | | | | | | | | | 79.99 | |
| | 8 | | | 59.58 | 58.86 | 50.36 | 49.82 | | | | | 67.56 | 66.82 | | | | | | | | |
| | 9 | 9.40 | 9.65 | 71.11 | 70.69 | | | 14.12 | 13.87 | 25.86 | 24.74 | 72.74 | 71.68 | 31.60 | 30.79 | 56.58 | 55.70 | 65.70 | 65.87 | | |
| | 10 | | | | | | | | | | | | | | | | | | | 64.29 | 63.50 |
| | 11 | | | 67.25 | 66.45 | 45.92 | 44.34 | | | | | 41.47 | 40.12 | 69.16 | 68.82 | 48.86 | 48.04 | 36.66 | 36.79 | | |
| | 12 | 50.50 | 50.83 | | | | | 69.58 | 69.84 | 71.31 | 71.79 | | | | | | | | | | |
| | 13 | | | | | | | | | | | | | | | | | | | 42.20 | 41.88 |
| | 14 | | | | | | | 67.61 | 67.72 | | | | | 75.07 | 74.70 | | | 65.82 | 65.03 | | |
| | 15 | 47.38 | 46.89 | 55.95 | 55.74 | 79.26 | 78.72 | 62.90 | 62.89 | 44.61 | 43.84 | 76.75 | 76.72 | 63.29 | 62.85 | 72.17 | 71.75 | | | | |
| | 16 | | | | | | | | | | | | | | | | | | | 52.36 | 51.80 |
| | 17 | | | 10.18 | 9.84 | 27.28 | 26.74 | 80.05 | 79.45 | 56.79 | 56.10 | 35.92 | 35.77 | 63.32 | 62.11 | 47.93 | 47.81 | 69.71 | 69.71 | | |
| | 18 | 30.87 | 30.75 | | | | | 33.92 | 34.80 | 56.11 | 55.74 | | | 56.27 | 56.77 | 79.50 | | 79.98 | | | |
| | 19 | | | | | | | | | | | | | | | 79.92 | 79.41 | 57.34 | 56.85 | | |
| | 20 | | | | | | | | | | | | | | | | | | | | |
| | 21 | | | | | | | | | | | | | | | | | | | | |
| | 22 | 51.06 | 50.70 | | | | | 34.97 | 34.73 | 60.35 | 59.70 | | | 48.20 | 47.73 | 66.47 | 66.65 | 56.14 | 55.40 | | |
| | 23 | | | | | | | | | | | | | | | | | | | 54.47 | 53.84 |
| | 24 | | | | | | | | | | | | | | | | | | | 62.21 | 61.69 |
| | 25 | | | | | | | | | | | | | | | | | | | 64.93 | 63.88 |
| | 26 | 69.31 | 66.89 | 55.28 | 54.76 | 77.06 | 76.74 | | | 62.89 | 59.85 | 80.35 | 78.74 | 79.58 | 76.86 | | | | | | |
| | 27 | | | 29.47 | 31.85 | 34.85 | 35.78 | | | 72.81 | 74.05 | 49.48 | 50.79 | 79.50 | | 63.82 | 65.82 | | | | |
| | 28 | | | 31.83 | 32.34 | 8.77 | 8.56 | 79.04 | 79.74 | 63.15 | 63.45 | 13.41 | 13.85 | 58.73 | 59.46 | 33.80 | 34.85 | 46.72 | 46.73 | | |
| | 29 | | | | | | | | | | | | | | | | | | | 45.04 | 44.69 |
| | 30 | 80.77 | 79.80 | | | | | | | | | | | | | | | | | | |
| | 31 | | | | | | | | | | | | | | | | | | | 78.80 | 77.75 |
| | 32 | | | | | | | | | | | | | | | | | 72.61 | 73.82 | | |

# APPENDIX G. COMPARISON OF DOP VALUES FROM OUTDATED ALMANAC DATA

| Time | Almanac Used- Date: Jul 28, 2008; Week: 466; Time of Applicability: 319488 Jul 29, 2008 (1-day old almanac) | | | | | | | | Almanac Used- Date: Jul 29, 2008; Week: 466; Time of Applicability: 405504 Jul 29, 2008 (current almanac) | | | | | | | | Percentage Difference | | | | | | | Difference in Visible SV |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | |
| 0:00 | 1.675 | 1.496 | 0.862 | 0.755 | 1.222 | 0.477 | 0.719 | 11 | 1.675 | 1.496 | 0.862 | 0.755 | 1.222 | 0.477 | 0.719 | 11 | 0.003 | 0.003 | 0.001 | 0.003 | 0.003 | 0.001 | 0.001 | 0 |
| 0:10 | 1.834 | 1.624 | 0.933 | 0.852 | 1.329 | 0.529 | 0.769 | 10 | 1.834 | 1.624 | 0.933 | 0.852 | 1.329 | 0.529 | 0.769 | 10 | 0.003 | 0.003 | 0.002 | 0.003 | 0.004 | 0.001 | 0.002 | 0 |
| 0:20 | 2.101 | 1.846 | 0.991 | 1.002 | 1.558 | 0.562 | 0.816 | 9 | 2.100 | 1.846 | 0.991 | 1.002 | 1.558 | 0.562 | 0.816 | 9 | 0.002 | 0.003 | 0.002 | 0.002 | 0.003 | 0.000 | 0.004 | 0 |
| 0:40 | 2.331 | 2.058 | 1.210 | 1.095 | 1.664 | 0.582 | 1.062 | 8 | 2.331 | 2.058 | 1.210 | 1.095 | 1.664 | 0.582 | 1.062 | 8 | 0.003 | 0.003 | 0.004 | 0.003 | 0.003 | 0.002 | 0.005 | 0 |
| 1:00 | 2.742 | 2.388 | 1.281 | 1.346 | 2.016 | 0.602 | 1.131 | 7 | 2.742 | 2.388 | 1.281 | 1.346 | 2.016 | 0.602 | 1.131 | 7 | 0.004 | 0.004 | 0.005 | 0.004 | 0.004 | 0.001 | 0.006 | 0 |
| 2:00 | 3.025 | 2.626 | 1.265 | 1.502 | 2.301 | 0.650 | 1.085 | 6 | 3.024 | 2.625 | 1.265 | 1.501 | 2.301 | 0.650 | 1.085 | 6 | 0.004 | 0.004 | 0.003 | 0.004 | 0.005 | -0.001 | 0.004 | 0 |
| 4:00 | 1.854 | 1.700 | 0.716 | 0.741 | 1.541 | 0.452 | 0.556 | 11 | 1.854 | 1.700 | 0.716 | 0.741 | 1.541 | 0.452 | 0.556 | 11 | 0.001 | 0.000 | 0.001 | 0.001 | 0.000 | -0.001 | 0.001 | 0 |
| 6:00 | 1.532 | 1.385 | 0.715 | 0.655 | 1.186 | 0.487 | 0.524 | 12 | 1.532 | 1.385 | 0.715 | 0.655 | 1.186 | 0.487 | 0.524 | 12 | 0.003 | 0.003 | 0.000 | 0.002 | 0.004 | 0.002 | 0.000 | 0 |
| 10:00 | 1.889 | 1.697 | 0.844 | 0.829 | 1.472 | 0.529 | 0.657 | 10 | 1.889 | 1.697 | 0.844 | 0.829 | 1.472 | 0.530 | 0.657 | 10 | -0.001 | -0.001 | 0.000 | -0.001 | -0.002 | -0.002 | 0.001 | 0 |
| 14:00 | 1.984 | 1.791 | 0.943 | 0.852 | 1.523 | 0.565 | 0.755 | 9 | 1.984 | 1.791 | 0.943 | 0.852 | 1.523 | 0.565 | 0.756 | 9 | 0.004 | 0.004 | -0.002 | 0.004 | 0.006 | 0.002 | -0.005 | 0 |
| 18:00 | 2.505 | 2.173 | 1.160 | 1.245 | 1.838 | 0.703 | 0.922 | 8 | 2.505 | 2.173 | 1.160 | 1.245 | 1.838 | 0.703 | 0.922 | 8 | 0.001 | 0.001 | 0.000 | 0.002 | 0.002 | -0.002 | 0.001 | 0 |
| 22:00 | 1.959 | 1.715 | 0.948 | 0.948 | 1.429 | 0.584 | 0.747 | 10 | 1.959 | 1.715 | 0.948 | 0.948 | 1.429 | 0.584 | 0.747 | 10 | -0.002 | -0.002 | -0.001 | -0.003 | -0.002 | 0.001 | -0.002 | 0 |
| 23:00 | 1.859 | 1.618 | 0.899 | 0.914 | 1.346 | 0.561 | 0.702 | 10 | 1.859 | 1.618 | 0.899 | 0.914 | 1.346 | 0.561 | 0.702 | 10 | -0.001 | -0.001 | 0.000 | -0.001 | -0.001 | 0.001 | -0.001 | 0 |
| 23:50 | 1.648 | 1.470 | 0.847 | 0.745 | 1.201 | 0.479 | 0.699 | 11 | 1.648 | 1.470 | 0.847 | 0.745 | 1.201 | 0.479 | 0.699 | 11 | 0.002 | 0.001 | 0.001 | 0.002 | 0.002 | 0.000 | 0.001 | 0 |

| Time | Almanac Used- Date: Jul 28, 2008; Week: 466; Time of Applicability: 319488 Aug 4, 2008 (7-day old almanac) | | | | | | | | Almanac Used- Date: Jul 29, 2008; Week: 467; Time of Applicability: 319488 Aug 4, 2008 (current almanac) | | | | | | | | Percentage Difference | | | | | | | Difference in Visible SV |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | |
| 0:00 | 2.120 | 1.864 | 1.006 | 1.009 | 1.569 | 0.557 | 0.838 | 9 | 2.122 | 1.866 | 1.008 | 1.010 | 1.570 | 0.556 | 0.841 | 9 | -0.092 | -0.090 | -0.161 | -0.101 | -0.060 | 0.135 | -0.291 | 0 |
| 0:10 | 2.161 | 1.905 | 1.048 | 1.020 | 1.591 | 0.542 | 0.897 | 9 | 2.164 | 1.908 | 1.050 | 1.022 | 1.592 | 0.542 | 0.900 | 9 | -0.125 | -0.120 | -0.201 | -0.139 | -0.085 | 0.114 | -0.315 | 0 |
| 0:20 | 2.826 | 2.445 | 1.255 | 1.417 | 2.098 | 0.615 | 1.094 | 7 | 2.826 | 2.445 | 1.257 | 1.417 | 2.098 | 0.614 | 1.097 | 7 | -0.008 | -0.023 | -0.135 | 0.037 | 0.018 | 0.147 | -0.224 | 0 |
| 0:40 | 2.710 | 2.365 | 1.288 | 1.322 | 1.984 | 0.601 | 1.140 | 7 | 2.712 | 2.368 | 1.290 | 1.323 | 1.985 | 0.600 | 1.142 | 7 | -0.090 | -0.092 | -0.145 | -0.085 | -0.070 | 0.138 | -0.223 | 0 |
| 1:00 | 2.222 | 1.972 | 1.129 | 1.023 | 1.617 | 0.600 | 0.957 | 8 | 2.223 | 1.973 | 1.130 | 1.024 | 1.618 | 0.600 | 0.958 | 8 | -0.046 | -0.047 | -0.069 | -0.043 | -0.036 | 0.007 | -0.099 | 0 |
| 2:00 | 2.270 | 2.019 | 1.009 | 1.038 | 1.749 | 0.608 | 0.805 | 7 | 2.266 | 2.015 | 1.006 | 1.036 | 1.746 | 0.608 | 0.802 | 7 | 0.182 | 0.176 | 0.219 | 0.207 | 0.161 | -0.068 | 0.384 | 0 |
| 4:00 | 1.767 | 1.619 | 0.705 | 0.708 | 1.458 | 0.465 | 0.530 | 11 | 1.767 | 1.619 | 0.705 | 0.708 | 1.458 | 0.465 | 0.530 | 11 | 0.022 | 0.006 | 0.017 | 0.106 | 0.004 | -0.016 | 0.042 | 0 |
| 6:00 | 1.993 | 1.758 | 0.853 | 0.938 | 1.537 | 0.595 | 0.612 | 11 | 1.993 | 1.758 | 0.853 | 0.938 | 1.537 | 0.595 | 0.612 | 11 | -0.002 | -0.004 | 0.002 | 0.004 | -0.006 | -0.005 | 0.008 | 0 |
| 10:00 | 1.728 | 1.573 | 0.806 | 0.714 | 1.350 | 0.545 | 0.594 | 11 | 1.727 | 1.573 | 0.806 | 0.715 | 1.350 | 0.545 | 0.594 | 11 | 0.001 | 0.003 | 0.005 | -0.008 | 0.002 | 0.001 | 0.009 | 0 |
| 14:00 | 2.035 | 1.843 | 1.070 | 0.863 | 1.501 | 0.579 | 0.900 | 8 | 2.034 | 1.843 | 1.070 | 0.862 | 1.500 | 0.579 | 0.900 | 8 | 0.039 | 0.034 | 0.018 | 0.063 | 0.043 | 0.011 | 0.020 | 0 |
| 18:00 | 2.138 | 1.889 | 1.037 | 1.000 | 1.579 | 0.617 | 0.833 | 10 | 2.138 | 1.889 | 1.037 | 1.000 | 1.579 | 0.617 | 0.833 | 10 | 0.001 | -0.001 | -0.021 | 0.010 | 0.008 | 0.023 | -0.045 | 0 |
| 22:00 | 1.791 | 1.578 | 0.882 | 0.847 | 1.309 | 0.555 | 0.685 | 11 | 1.790 | 1.577 | 0.882 | 0.847 | 1.308 | 0.555 | 0.686 | 11 | 0.019 | 0.021 | -0.099 | 0.013 | 0.075 | -0.048 | 0.132 | 0 |
| 23:00 | 1.941 | 1.693 | 0.953 | 0.951 | 1.399 | 0.551 | 0.777 | 10 | 1.944 | 1.695 | 0.955 | 0.952 | 1.400 | 0.550 | 0.780 | 10 | -0.128 | -0.121 | -0.235 | -0.150 | -0.067 | 0.108 | -0.406 | 0 |
| 23:50 | 2.084 | 1.831 | 0.984 | 0.995 | 1.544 | 0.565 | 0.806 | 9 | 2.086 | 1.833 | 0.986 | 0.996 | 1.545 | 0.564 | 0.809 | 9 | -0.083 | -0.081 | -0.156 | -0.090 | -0.051 | 0.171 | -0.315 | 0 |

| Almanac Used- Date: Jul 28, 2008; Week: 466; Time of Applicability: 319488 Aug 11, 2008 (14-day old almanac) | | | | | | | | | Almanac Used- Date: Aug 11, 2008; Week: 468; Time of Applicability: 319488 Aug 11, 2008 (current almanac) | | | | | | | | Percentage Difference | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Time | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | Difference in Visible SV |
| 0:00 | 2.769 | 2.404 | 1.265 | 1.373 | 2.045 | 0.606 | 1.111 | 7 | 2.770 | 2.406 | 1.267 | 1.374 | 2.045 | 0.604 | 1.114 | 7 | -0.037 | -0.046 | -0.150 | -0.008 | -0.006 | 0.290 | -0.280 | 0 |
| 0:10 | 2.713 | 2.366 | 1.284 | 1.327 | 1.988 | 0.601 | 1.134 | 7 | 2.716 | 2.369 | 1.286 | 1.328 | 1.990 | 0.599 | 1.137 | 7 | -0.115 | -0.108 | -0.147 | -0.135 | -0.092 | 0.258 | -0.259 | 0 |
| 0:20 | 2.142 | 1.900 | 1.106 | 0.990 | 1.545 | 0.587 | 0.937 | 8 | 2.141 | 1.899 | 1.106 | 0.989 | 1.544 | 0.586 | 0.938 | 8 | 0.045 | 0.033 | -0.020 | 0.092 | 0.060 | 0.040 | -0.043 | 0 |
| 0:40 | 2.423 | 2.140 | 1.197 | 1.137 | 1.774 | 0.620 | 1.024 | 7 | 2.423 | 2.139 | 1.197 | 1.137 | 1.773 | 0.621 | 1.023 | 7 | 0.006 | 0.002 | -0.029 | 0.020 | 0.016 | -0.208 | 0.037 | 0 |
| 1:00 | 2.556 | 2.259 | 1.222 | 1.196 | 1.900 | 0.625 | 1.050 | 7 | 2.559 | 2.261 | 1.222 | 1.198 | 1.903 | 0.626 | 1.049 | 7 | -0.120 | -0.095 | 0.006 | -0.209 | -0.137 | -0.239 | 0.093 | 0 |
| 2:00 | 2.012 | 1.820 | 0.887 | 0.857 | 1.589 | 0.546 | 0.699 | 8 | 2.006 | 1.815 | 0.884 | 0.854 | 1.586 | 0.547 | 0.694 | 8 | 0.260 | 0.243 | 0.352 | 0.337 | 0.209 | -0.212 | 0.701 | 0 |
| 4:00 | 1.907 | 1.713 | 0.824 | 0.840 | 1.502 | 0.551 | 0.612 | 10 | 1.897 | 1.705 | 0.823 | 0.833 | 1.493 | 0.551 | 0.611 | 10 | 0.541 | 0.466 | 0.109 | 0.853 | 0.574 | -0.015 | 0.209 | 0 |
| 6:00 | 1.955 | 1.732 | 0.818 | 0.907 | 1.526 | 0.568 | 0.589 | 11 | 1.955 | 1.732 | 0.818 | 0.907 | 1.527 | 0.568 | 0.589 | 11 | -0.023 | -0.025 | 0.006 | -0.016 | -0.033 | 0.031 | -0.017 | 0 |
| 10:00 | 1.731 | 1.577 | 0.703 | 0.714 | 1.411 | 0.483 | 0.511 | 12 | 1.730 | 1.576 | 0.703 | 0.714 | 1.411 | 0.483 | 0.511 | 12 | 0.042 | 0.043 | 0.014 | 0.037 | 0.050 | -0.013 | 0.038 | 0 |
| 14:00 | 2.065 | 1.874 | 1.017 | 0.866 | 1.575 | 0.502 | 0.884 | 8 | 2.065 | 1.874 | 1.017 | 0.866 | 1.574 | 0.502 | 0.884 | 8 | 0.018 | 0.020 | 0.014 | 0.009 | 0.023 | 0.009 | 0.016 | 0 |
| 18:00 | 2.532 | 2.221 | 1.132 | 1.216 | 1.910 | 0.537 | 0.997 | 9 | 2.532 | 2.221 | 1.133 | 1.216 | 1.910 | 0.537 | 0.998 | 9 | -0.020 | -0.027 | -0.096 | 0.005 | -0.003 | 0.065 | -0.142 | 0 |
| 22:00 | 1.682 | 1.484 | 0.871 | 0.793 | 1.201 | 0.534 | 0.688 | 11 | 1.685 | 1.486 | 0.874 | 0.795 | 1.202 | 0.534 | 0.692 | 11 | -0.163 | -0.161 | -0.361 | -0.170 | -0.056 | 0.054 | -0.610 | 0 |
| 23:00 | 1.649 | 1.472 | 0.853 | 0.744 | 1.199 | 0.479 | 0.706 | 11 | 1.652 | 1.474 | 0.856 | 0.745 | 1.200 | 0.478 | 0.711 | 11 | -0.145 | -0.138 | -0.319 | -0.169 | -0.047 | 0.259 | -0.581 | 0 |
| 23:50 | 2.801 | 2.426 | 1.257 | 1.400 | 2.074 | 0.612 | 1.099 | 7 | 2.800 | 2.426 | 1.259 | 1.399 | 2.073 | 0.610 | 1.102 | 7 | 0.013 | -0.006 | -0.151 | 0.069 | 0.047 | 0.317 | -0.294 | 0 |

| Almanac Used- Date: Jul 28, 2008; Week: 466; Time of Applicability: 319488 Aug 27, 2008 (30-day old almanac) | | | | | | | | | Almanac Used- Date: Aug 27, 2008; Week: 470; Time of Applicability: 503808 Aug 27, 2008 (current almanac) | | | | | | | | Percentage Difference | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Time | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | No. of Visible SV | GDOP | PDOP | HDOP | TDOP | VDOP | YDOP | XDOP | Difference in Visible SV |
| 0:00 | 3.024 | 2.626 | 1.268 | 1.500 | 2.300 | 0.647 | 1.090 | 6 | 3.016 | 2.619 | 1.260 | 1.496 | 2.297 | 0.652 | 1.078 | 6 | 0.263 | 0.256 | 0.606 | 0.286 | 0.150 | -0.802 | 1.117 | 0 |
| 0:10 | 3.145 | 2.728 | 1.260 | 1.564 | 2.419 | 0.634 | 1.089 | 6 | 3.145 | 2.728 | 1.253 | 1.565 | 2.423 | 0.639 | 1.078 | 6 | -0.006 | 0.008 | 0.592 | -0.049 | -0.149 | -0.702 | 1.042 | 0 |
| 0:20 | 2.288 | 2.031 | 1.024 | 1.053 | 1.754 | 0.602 | 0.829 | 7 | 2.269 | 2.016 | 1.010 | 1.042 | 1.745 | 0.604 | 0.809 | 7 | 0.800 | 0.745 | 1.408 | 1.005 | 0.521 | -0.412 | 2.410 | 0 |
| 0:40 | 2.070 | 1.864 | 0.913 | 0.900 | 1.626 | 0.543 | 0.733 | 8 | 2.055 | 1.852 | 0.902 | 0.891 | 1.617 | 0.546 | 0.718 | 8 | 0.753 | 0.686 | 1.206 | 1.044 | 0.524 | -0.413 | 2.128 | 0 |
| 1:00 | 1.995 | 1.807 | 0.881 | 0.845 | 1.578 | 0.543 | 0.693 | 8 | 1.990 | 1.803 | 0.874 | 0.841 | 1.577 | 0.546 | 0.683 | 8 | 0.265 | 0.216 | 0.754 | 0.489 | 0.050 | -0.492 | 1.541 | 0 |
| 2:00 | 1.836 | 1.682 | 0.718 | 0.736 | 1.521 | 0.453 | 0.558 | 11 | 1.825 | 1.674 | 0.714 | 0.727 | 1.514 | 0.454 | 0.552 | 11 | 0.589 | 0.482 | 0.600 | 1.153 | 0.455 | -0.140 | 1.097 | 0 |
| 4:00 | 1.550 | 1.400 | 0.716 | 0.665 | 1.202 | 0.486 | 0.526 | 12 | 1.897 | 1.677 | 0.856 | 0.888 | 1.442 | 0.554 | 0.653 | 11 | -18.334 | -16.532 | -16.367 | -25.108 | -16.590 | -12.189 | -19.503 | 1 |
| 6:00 | 1.815 | 1.636 | 0.701 | 0.785 | 1.478 | 0.458 | 0.531 | 12 | 1.815 | 1.636 | 0.701 | 0.786 | 1.478 | 0.458 | 0.531 | 12 | -0.029 | -0.031 | -0.025 | -0.022 | -0.032 | 0.048 | -0.079 | 0 |
| 10:00 | 2.363 | 2.103 | 0.825 | 1.078 | 1.935 | 0.609 | 0.557 | 9 | 2.363 | 2.103 | 0.825 | 1.078 | 1.934 | 0.609 | 0.556 | 9 | 0.021 | 0.019 | 0.014 | 0.029 | 0.019 | -0.056 | 0.099 | 0 |
| 14:00 | 1.893 | 1.710 | 0.800 | 0.811 | 1.511 | 0.489 | 0.633 | 10 | 1.889 | 1.707 | 0.800 | 0.810 | 1.508 | 0.489 | 0.633 | 10 | 0.177 | 0.183 | -0.001 | 0.148 | 0.235 | -0.091 | 0.053 | 0 |
| 18:00 | 1.714 | 1.549 | 0.910 | 0.734 | 1.253 | 0.440 | 0.797 | 11 | 1.714 | 1.549 | 0.909 | 0.735 | 1.253 | 0.440 | 0.796 | 11 | -0.009 | 0.013 | 0.073 | -0.105 | -0.019 | -0.086 | 0.121 | 0 |
| 22:00 | 1.656 | 1.479 | 0.866 | 0.744 | 1.200 | 0.478 | 0.722 | 11 | 1.662 | 1.484 | 0.872 | 0.747 | 1.201 | 0.475 | 0.731 | 11 | -0.352 | -0.325 | -0.729 | -0.457 | -0.113 | 0.539 | -1.270 | 0 |
| 23:00 | 2.703 | 2.357 | 1.278 | 1.324 | 1.980 | 0.601 | 1.128 | 7 | 2.704 | 2.357 | 1.278 | 1.325 | 1.980 | 0.598 | 1.130 | 7 | -0.035 | -0.006 | -0.002 | -0.126 | -0.008 | 0.458 | -0.131 | 0 |
| 23:50 | 2.560 | 2.263 | 1.218 | 1.199 | 1.907 | 0.621 | 1.048 | 7 | 2.563 | 2.264 | 1.214 | 1.202 | 1.910 | 0.625 | 1.042 | 7 | -0.100 | -0.043 | 0.295 | -0.302 | -0.180 | -0.571 | 0.604 | 0 |

# APPENDIX H. AVERAGE DOP VALUES

| Date | | Jul 28, 2008 | Jul 31, 2008 | Aug 11, 2008 | Aug 27, 2008 | Dec 3, 2008 | Feb 3, 2009 | **Global Average** |
|---|---|---|---|---|---|---|---|---|
| **Average DOP for Obstruction Angle = 0°** | **XDOP** | 0.516 | 0.516 | 0.523 | 0.529 | 0.514 | 0.515 | **0.519** |
| | **YDOP** | 0.55 | 0.55 | 0.558 | 0.565 | 0.548 | 0.549 | **0.553** |
| | **VDOP** | 1.202 | 1.201 | 1.221 | 1.242 | 1.194 | 1.197 | **1.210** |
| | **TDOP** | 0.649 | 0.649 | 0.66 | 0.672 | 0.644 | 0.646 | **0.653** |
| | **HDOP** | 0.758 | 0.758 | 0.768 | 0.777 | 0.755 | 0.757 | **0.762** |
| | **PDOP** | 1.426 | 1.426 | 1.447 | 1.47 | 1.417 | 1.421 | **1.435** |
| | **GDOP** | 1.567 | 1.567 | 1.591 | 1.617 | 1.557 | 1.562 | **1.577** |
| **Average DOP for Obstruction Angle = 10°** | **XDOP** | 0.637 | 0.636 | 0.636 | 0.635 | 0.632 | 0.633 | **0.635** |
| | **YDOP** | 0.715 | 0.714 | 0.713 | 0.711 | 0.71 | 0.711 | **0.712** |
| | **VDOP** | 1.772 | 1.77 | 1.767 | 1.762 | 1.758 | 1.759 | **1.765** |
| | **TDOP** | 1.08 | 1.079 | 1.076 | 1.073 | 1.071 | 1.071 | **1.075** |
| | **HDOP** | 0.966 | 0.966 | 0.964 | 0.962 | 0.959 | 0.96 | **0.963** |
| | **PDOP** | 2.029 | 2.027 | 2.023 | 2.017 | 2.013 | 2.014 | **2.021** |
| | **GDOP** | 2.3 | 2.298 | 2.293 | 2.287 | 2.282 | 2.283 | **2.291** |
| **Average DOP for Obstruction Angle = 15°** | **XDOP** | 0.732 | 0.735 | 0.743 | 0.756 | 0.725 | 0.721 | **0.735** |
| | **YDOP** | 0.885 | 0.89 | 0.894 | 0.918 | 0.865 | 0.853 | **0.884** |
| | **VDOP** | 2.361 | 2.393 | 2.417 | 2.512 | 2.319 | 2.262 | **2.377** |
| | **TDOP** | 1.542 | 1.564 | 1.581 | 1.65 | 1.508 | 1.47 | **1.553** |
| | **HDOP** | 1.169 | 1.175 | 1.182 | 1.209 | 1.145 | 1.132 | **1.169** |
| | **PDOP** | 2.654 | 2.686 | 2.712 | 2.811 | 2.605 | 2.546 | **2.669** |
| | **GDOP** | 3.072 | 3.111 | 3.143 | 3.263 | 3.014 | 2.962 | **3.094** |

THIS PAGE INTENTIONALLY LEFT BLANK

# APPENDIX I. COMPARISON BETWEEN VDOP & HDOP FOR OBSTRUCTION ANGLE = $10^{O}$

| Lat (+/-90) (+North/-South) | From Matlab Program | Long (+/-180) (+East/-West) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | -165 | -150 | -135 | -120 | -105 | -90 | -75 | -60 | -45 | -30 | -15 | 0 |
| | HDOP | 1.0625 | 0.9999 | 0.9639 | 0.7899 | 0.7551 | 0.7345 | 0.7625 | 0.7765 | 0.8121 | 0.8432 | 0.9372 | 0.9481 |
| -75 | VDOP | 2.2301 | 2.1614 | 2.1693 | 1.6064 | 1.6105 | 1.6104 | 1.8262 | 1.815 | 1.8024 | 1.8412 | 2.081 | 2.0896 |
| | **VDOP/HDOP** | **2.10** | **2.16** | **2.25** | **2.03** | **2.13** | **2.19** | **2.40** | **2.34** | **2.22** | **2.18** | **2.22** | **2.20** |
| | HDOP | 0.9585 | 0.8182 | 0.7652 | 0.88 | 0.8897 | 0.9376 | 0.7838 | 0.8109 | 0.8257 | 0.7788 | 1.0127 | 1.1056 |
| -60 | VDOP | 1.5495 | 1.4715 | 1.399 | 1.7528 | 1.7527 | 1.8916 | 1.3482 | 1.3835 | 1.3694 | 1.1682 | 1.2553 | 1.3884 |
| | **VDOP/HDOP** | **1.62** | **1.80** | **1.83** | **1.99** | **1.97** | **2.02** | **1.72** | **1.71** | **1.66** | **1.50** | **1.24** | **1.26** |
| | HDOP | 0.7845 | 1.0444 | 0.8567 | 0.8024 | 0.9586 | 1.1863 | 1.0722 | 0.9991 | 0.9872 | 0.9932 | 1.2276 | 1.1133 |
| -45 | VDOP | 1.226 | 1.9465 | 1.719 | 1.5231 | 1.6833 | 1.5174 | 1.6146 | 1.339 | 1.3713 | 1.3789 | 1.877 | 1.6956 |
| | **VDOP/HDOP** | **1.56** | **1.86** | **2.01** | **1.90** | **1.76** | **1.28** | **1.51** | **1.34** | **1.39** | **1.39** | **1.53** | **1.52** |
| | HDOP | 0.8891 | 1.1133 | 0.94 | 0.8012 | 0.9545 | 1.126 | 1.0938 | 1.0556 | 1.0681 | 1.1437 | 0.8942 | 1.0076 |
| -30 | VDOP | 1.5539 | 1.6167 | 1.5316 | 1.3761 | 1.5711 | 1.5216 | 1.3059 | 1.5371 | 1.5207 | 1.8955 | 1.5889 | 2.5775 |
| | **VDOP/HDOP** | **1.75** | **1.45** | **1.63** | **1.72** | **1.65** | **1.35** | **1.19** | **1.46** | **1.42** | **1.66** | **1.78** | **2.56** |
| | HDOP | 0.8969 | 0.9465 | 0.8757 | 1.0302 | 0.9421 | 0.7521 | 1.0238 | 0.9769 | 1.0051 | 0.9153 | 0.7738 | 0.9584 |
| -15 | VDOP | 1.6166 | 1.5739 | 1.3769 | 1.6733 | 1.5746 | 1.3061 | 1.7545 | 1.5795 | 1.5527 | 1.4673 | 1.321 | 2.6471 |
| | **VDOP/HDOP** | **1.80** | **1.66** | **1.57** | **1.62** | **1.67** | **1.74** | **1.71** | **1.62** | **1.54** | **1.60** | **1.71** | **2.76** |
| | HDOP | 0.8261 | 0.9784 | 0.8688 | 0.9202 | 0.7573 | 0.758 | 0.8037 | 0.9542 | 0.7587 | 0.8196 | 0.8464 | 0.8627 |
| 0 | VDOP | 1.2304 | 1.5315 | 1.4217 | 1.845 | 1.4634 | 1.479 | 1.6135 | 1.6283 | 1.0743 | 1.3739 | 1.3908 | 1.3948 |
| | **VDOP/HDOP** | **1.49** | **1.57** | **1.64** | **2.00** | **1.93** | **1.95** | **2.01** | **1.71** | **1.42** | **1.68** | **1.64** | **1.62** |
| | HDOP | 1.0409 | 0.8134 | 0.9886 | 0.9577 | 0.981 | 0.7759 | 0.8367 | 0.7649 | 0.9597 | 0.7841 | 0.9984 | 0.8391 |
| 15 | VDOP | 1.4624 | 1.0984 | 1.6328 | 2.3865 | 2.4136 | 1.5255 | 2.0938 | 1.3559 | 1.7049 | 1.2506 | 1.756 | 1.2575 |
| | **VDOP/HDOP** | **1.40** | **1.35** | **1.65** | **2.49** | **2.46** | **1.97** | **2.50** | **1.77** | **1.78** | **1.59** | **1.76** | **1.50** |
| | HDOP | 1.0529 | 1.1707 | 0.8603 | 0.9373 | 0.8881 | 0.9799 | 1.1539 | 0.9204 | 0.8318 | 0.9465 | 0.8983 | 1.153 |
| 30 | VDOP | 1.8536 | 1.8373 | 1.3208 | 1.6216 | 1.6382 | 1.5464 | 2.3453 | 1.6779 | 1.2521 | 1.5605 | 1.4153 | 1.7551 |
| | **VDOP/HDOP** | **1.76** | **1.57** | **1.54** | **1.73** | **1.84** | **1.58** | **2.03** | **1.82** | **1.51** | **1.65** | **1.58** | **1.52** |
| | HDOP | 0.8857 | 1.1367 | 0.9718 | 1.0757 | 1.1747 | 1.211 | 1.8173 | 1.1202 | 1.1452 | 1.2925 | 0.9871 | 0.8529 |
| 45 | VDOP | 1.6452 | 2.7577 | 1.5247 | 1.5603 | 1.5268 | 1.3642 | 2.6174 | 1.5663 | 1.5405 | 2.3618 | 1.9303 | 1.5966 |
| | **VDOP/HDOP** | **1.86** | **2.43** | **1.57** | **1.45** | **1.30** | **1.13** | **1.44** | **1.40** | **1.35** | **1.83** | **1.96** | **1.87** |
| | HDOP | 0.8329 | 0.7672 | 0.7985 | 1.0052 | 1.81 | 1.9523 | 1.503 | 1.5491 | 0.955 | 0.7997 | 0.919 | 0.8921 |
| 60 | VDOP | 1.6323 | 1.5154 | 1.4979 | 1.7956 | 3.2854 | 3.192 | 1.258 | 1.6089 | 1.3477 | 1.3313 | 1.7181 | 1.9745 |
| | **VDOP/HDOP** | **1.96** | **1.98** | **1.88** | **1.79** | **1.82** | **1.63** | **0.84** | **1.04** | **1.41** | **1.66** | **1.87** | **2.21** |
| | HDOP | 0.7976 | 0.8176 | 1.0087 | 1.0545 | 1.1446 | 1.1119 | 0.9453 | 0.9397 | 0.9897 | 0.9589 | 0.8009 | 0.7692 |
| 75 | VDOP | 2.0214 | 2.0239 | 3.6001 | 3.5553 | 3.5077 | 2.6718 | 2.0048 | 2.0016 | 2.3267 | 2.3437 | 1.7415 | 1.7577 |
| | **VDOP/HDOP** | **2.53** | **2.48** | **3.57** | **3.37** | **3.06** | **2.40** | **2.12** | **2.13** | **2.35** | **2.44** | **2.17** | **2.29** |
| | HDOP | | | | | | | | | | | | 0.7752 |
| 90 | VDOP | | | | | | | | | | | | 2.0945 |
| | **VDOP/HDOP** | | | | | | | | | | | | **2.70** |
| | HDOP | | | | | | | | | | | | 0.7591 |
| -90 | VDOP | | | | | | | | | | | | 1.7431 |
| | **VDOP/HDOP** | | | | | | | | | | | | **2.30** |

| Lat (+/-90) (+North/-South) | From Matlab Program | Long (+/-180) (+East/-West) | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | -165 | -150 | -135 | -120 | -105 | -90 | -75 | -60 | -45 | -30 | -15 | 0 |
| | HDOP | 0.7944 | 0.8576 | 0.8083 | 0.7838 | 0.788 | 0.8201 | 0.875 | 0.9452 | 1.0232 | 1.1487 | 0.9071 | 0.9576 |
| -75 | VDOP | 1.716 | 2.1094 | 2.1104 | 2.106 | 2.0962 | 2.0816 | 2.0632 | 2.0425 | 2.0214 | 2.2583 | 1.6538 | 1.6753 |
| | **VDOP/HDOP** | **2.16** | **2.46** | **2.61** | **2.69** | **2.66** | **2.54** | **2.36** | **2.16** | **1.98** | **1.97** | **1.82** | **1.75** |
| | HDOP | 1.1659 | 1.1252 | 1.0991 | 1.0589 | 0.805 | 0.8183 | 0.8726 | 0.8966 | 0.9209 | 0.9718 | 0.8621 | 0.8341 |
| -60 | VDOP | 1.6762 | 1.5156 | 2.0696 | 2.0705 | 1.4313 | 1.4146 | 1.3852 | 1.3301 | 1.3101 | 1.2843 | 1.1146 | 1.1732 |
| | **VDOP/HDOP** | **1.44** | **1.35** | **1.88** | **1.96** | **1.78** | **1.73** | **1.59** | **1.48** | **1.42** | **1.32** | **1.29** | **1.41** |
| | HDOP | 0.8257 | 0.8862 | 0.8058 | 0.8757 | 1.0216 | 1.0521 | 1.0043 | 0.982 | 1.0391 | 1.1071 | 1.4274 | 0.9377 |
| -45 | VDOP | 1.287 | 1.357 | 1.2419 | 1.3952 | 1.7959 | 1.7727 | 1.4092 | 1.412 | 1.3863 | 1.369 | 2.3315 | 1.4488 |
| | **VDOP/HDOP** | **1.56** | **1.53** | **1.54** | **1.59** | **1.76** | **1.68** | **1.40** | **1.44** | **1.33** | **1.24** | **1.63** | **1.55** |
| | HDOP | 0.8592 | 0.9217 | 0.9002 | 0.8921 | 0.9812 | 1.0376 | 0.9853 | 0.8617 | 0.9491 | 1.1597 | 0.9738 | 0.7794 |
| -30 | VDOP | 1.7422 | 1.3812 | 1.4426 | 1.767 | 1.7247 | 1.4541 | 1.4776 | 1.2248 | 1.5522 | 2.4277 | 2.0797 | 1.413 |
| | **VDOP/HDOP** | **2.03** | **1.50** | **1.60** | **1.98** | **1.76** | **1.40** | **1.50** | **1.42** | **1.64** | **2.09** | **2.14** | **1.81** |
| | HDOP | 0.7592 | 0.9618 | 0.9231 | 0.773 | 0.9219 | 0.933 | 0.847 | 0.9502 | 0.8104 | 0.7251 | 0.7854 | 1.0624 |
| -15 | VDOP | 1.3859 | 1.896 | 2.1601 | 1.4265 | 1.5445 | 1.3396 | 1.1934 | 1.6841 | 1.3686 | 1.3572 | 1.6216 | 2.3856 |
| | **VDOP/HDOP** | **1.83** | **1.97** | **2.34** | **1.85** | **1.68** | **1.44** | **1.41** | **1.77** | **1.69** | **1.87** | **2.06** | **2.25** |
| | HDOP | 0.7275 | 0.8056 | 0.8379 | 0.7659 | 0.9582 | 0.8064 | 0.8565 | 0.7197 | 0.766 | 0.7801 | 0.8587 | 0.9118 |
| 0 | VDOP | 1.2525 | 1.5234 | 1.6099 | 1.3801 | 1.6441 | 1.1709 | 1.6742 | 1.4733 | 1.4483 | 1.3977 | 1.5548 | 1.4238 |
| | **VDOP/HDOP** | **1.72** | **1.89** | **1.92** | **1.80** | **1.72** | **1.45** | **1.95** | **2.05** | **1.89** | **1.79** | **1.81** | **1.56** |
| | HDOP | 0.8667 | 0.7395 | 1.0045 | 0.9866 | 0.7814 | 0.8438 | 0.9131 | 0.7395 | 0.9047 | 0.7803 | 1.0516 | 1.0147 |
| 15 | VDOP | 1.4213 | 1.1908 | 1.7332 | 1.7296 | 1.0833 | 1.3334 | 2.445 | 1.7692 | 1.8635 | 1.273 | 1.3512 | 1.5009 |
| | **VDOP/HDOP** | **1.64** | **1.61** | **1.73** | **1.75** | **1.39** | **1.58** | **2.68** | **2.39** | **2.06** | **1.63** | **1.28** | **1.48** |
| | HDOP | 1.1499 | 0.9721 | 1.0491 | 1.0291 | 0.9251 | 0.9652 | 1.2426 | 0.9292 | 1.0271 | 1.2077 | 1.1479 | 1.0817 |
| 30 | VDOP | 1.5707 | 1.266 | 1.514 | 1.4928 | 1.284 | 1.3687 | 2.6182 | 2.1491 | 2.082 | 1.6792 | 1.6341 | 1.6741 |
| | **VDOP/HDOP** | **1.37** | **1.30** | **1.44** | **1.45** | **1.39** | **1.42** | **2.11** | **2.31** | **2.03** | **1.39** | **1.42** | **1.55** |
| | HDOP | 0.8955 | 1.0091 | 1.0674 | 1.065 | 0.9418 | 1.4463 | 1.4629 | 1.311 | 1.3456 | 0.9346 | 0.9773 | 0.8766 |
| 45 | VDOP | 1.6025 | 1.3686 | 1.8086 | 1.7611 | 1.2625 | 1.7182 | 2.0918 | 2.1355 | 2.0968 | 1.1849 | 1.5796 | 1.6298 |
| | **VDOP/HDOP** | **1.79** | **1.36** | **1.69** | **1.65** | **1.34** | **1.19** | **1.43** | **1.63** | **1.56** | **1.27** | **1.62** | **1.86** |
| | HDOP | 0.785 | 0.9297 | 0.8529 | 0.9995 | 1.0755 | 1.2293 | 0.9602 | 0.8795 | 0.8616 | 0.838 | 0.8327 | 0.8349 |
| 60 | VDOP | 1.7413 | 2.2768 | 1.6408 | 1.8467 | 1.793 | 1.81 | 1.3778 | 1.3048 | 1.3145 | 1.3845 | 1.4293 | 1.4995 |
| | **VDOP/HDOP** | **2.22** | **2.45** | **1.92** | **1.85** | **1.67** | **1.47** | **1.43** | **1.48** | **1.53** | **1.65** | **1.72** | **1.80** |
| | HDOP | 0.8595 | 0.8575 | 0.9034 | 0.9877 | 0.8687 | 0.9206 | 1.0007 | 0.8332 | 0.8076 | 0.7956 | 0.8236 | 0.7999 |
| 75 | VDOP | 2.3278 | 2.3316 | 2.3235 | 2.3043 | 1.8947 | 1.9547 | 2.2594 | 1.745 | 1.7581 | 1.7713 | 1.999 | 2.0127 |
| | **VDOP/HDOP** | **2.71** | **2.72** | **2.57** | **2.33** | **2.18** | **2.12** | **2.26** | **2.09** | **2.18** | **2.23** | **2.43** | **2.52** |

# LIST OF REFERENCES


[1] P. H. Dana, "Global Positioning System Overview," *The Geographer's Craft Project, Department of Geography, The University of Colorado at Boulder*, May 1, 2000. [Online]. Available: http://www.colorado.edu/geography/gcraft/notes/gps/gps.html. [Accessed: Apr. 16, 2008].

[2] Wikipedia, "List of Refractive Indices," *Wikipedia*. [Online]. Available: http://en.wikipedia.org/wiki/List_of_indices_of_refraction. [Accessed: Mar. 13, 2009].

[3] Wikipedia, "Global Positioning System," *Wikipedia*. [Online]. Available: http://en.wikipedia.org/wiki/Gps. [Accessed: Oct. 26, 2008].

[4] Wikipedia, "Dilution of Precision (GPS)," *Wikipedia*. [Online]. Available: http://en.wikipedia.org/wiki/Dilution_of_precision_(GPS). [Accessed: Feb. 2, 2009].

[5] United States Coast Guard, "Definition of a SEM Almanac," *United States Coast Guard*. [Online]. Available: http://www.navcen.uscg.gov/GPS/gpssem.htm. [Accessed: Feb. 11, 2009].

[6] Eric Weisstein, "Julian Date," *Eric Weisstein's World of Astronomy*. [Online]. Available: http://scienceworld.wolfram.com/astronomy/JulianDate.html. [Accessed: Feb. 11, 2009].

[7] U.S. Naval Observatory Time Service Department, "GPS Week Number Rollover," *U.S. Naval Observatory*. [Online]. Available: http://tycho.usno.navy.mil/gps_week.html. [Accessed: Feb. 11, 2009].

[8] Interface Control Document, *Navstar GPS Space Segment/ Navigation User Interface*, ICD-GPS-200C, ARINC Research Corporation, Oct. 10, 1993, pp. 97-100.

[9] Wikipedia, "Geodetic System," *Wikipedia*. [Online]. Available: http://en.wikipedia.org/wiki/Geodetic_system. [Accessed: Feb. 3, 2008].

[10] Trimble, "Planning Software Download," *Trimble*. [Online]. Available: http://www.trimble.com/planningsoftware_ts.asp?Nav=Collection-8423. [Accessed: Dec. 12, 2007].

[11] J. A. Farrell and M. Barth, *The Global Positioning System & Inertial Navigation*. New York: McGraw-Hill, 1999, pp. 321-324.

[12] Wikipedia, "Leap Second," *Wikipedia*. [Online]. Available: http://en.wikipedia.org/wiki/Leap_second. [Accessed: Feb. 3, 2009].

[13] M. R. Driels, "Accuracy of GPS/ INS Guided Munitions," unpublished chapter for next edition of *Weaponeering: Conventional Weapon System Effectiveness.*

[14] M. R. Driels, *Weaponeering: Conventional Weapon System Effectiveness.* Reston, Virginia: American Institute of Aeronautics and Astronautics, Inc., 2004.

[15] United States Army, Corps of Engineers, *NAVSTAR Global Positioning System Surveying*. Reston, Virginia: American Society of Civil Engineers Press, 2000.

[16] Wikipedia, "GNSS Augmentation," *Wikipedia*. [Online]. Available: http://en.wikipedia.org/wiki/GNSS_Augmentation. [Accessed: Mar. 12, 2009].

[17] United States Coast Guard, "Navigation Center," *United States Coast Guard*. [Online]. Available: http://www.navcen.uscg.gov/gps/default.htm. [Accessed: Feb. 11, 2009].

[18] Weaponeering, "Weaponeering: Conventional Weapon Systems Effects," *Weaponeering*. [Online]. Available: http://www.weaponeering.com/. [Accessed: Feb. 12, 2009].

[19] Garmin, "What is GPS?," *Garmin*. [Online]. Available: http://www8.garmin.com/aboutGPS/. [Accessed: Oct. 26, 2008].

[20] Guestgulkan, "CGraph," *Codeguru*. [Online]. Available: http://www.codeguru.com/cpp/controls/controls/chartingandanalogguecontrols/article.php/c9309/. [Accessed: Dec. 4, 2008].

[21] A. Garcia, "Programs from Numerical Methods for Physics (Second Edition)," [Online]. Available: http://www.algarcia.org/nummeth/Programs2E.html. [Accessed: Oct. 1, 2008].

# INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, VA

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, CA

3. Ms. Carolyn Holand
   671 ARSS, Eglin AFB
   Fort Walton Beach, FL

4. Dr. Don Grundel
   671 ARSS, Eglin AFB
   Fort Walton Beach, FL

5. Dr. Mike Caluda
   671 ARSS, Eglin AFB
   Fort Walton Beach, FL

6. Ms. Teri Irwin
   Eglin AFB
   Fort Walton Beach, FL

7. Ms. Collen Rooney
   Eglin AFB
   Fort Walton Beach, FL

8. Mr. Robert Chandler
   US Army/ AMSAA
   Aberdeen, MD

9. Mr. Jim Matts
   US Army/ AMSAA
   Aberdeen, MD

10. Mr. Nate Kimbler
    NAWC
    China Lake, CA

11. Mr. Jerr Wyant
    NSAWC
    Fallon, NV

12. LT William Mansfield
Joint Targeting School
Virginia Beach, VA